Journal Pre-proof

Progressive growing of patch size: Curriculum learning for accelerated and improved medical image segmentation

Stefan M. Fischer, Johannes Kiechle, Laura Daza, Lina Felsner, Richard Osuala, Daniel M. Lang, Karim Lekadir, Jan C. Peeken, Julia A. Schnabel

PII: S1361-8415(26)00264-1
DOI: https://doi.org/10.1016/j.media.2026.104195
Reference: MEDIMA 104195

To appear in: *Medical Image Analysis*

Received date : 30 October 2025
Revised date : 15 June 2026
Accepted date : 23 June 2026

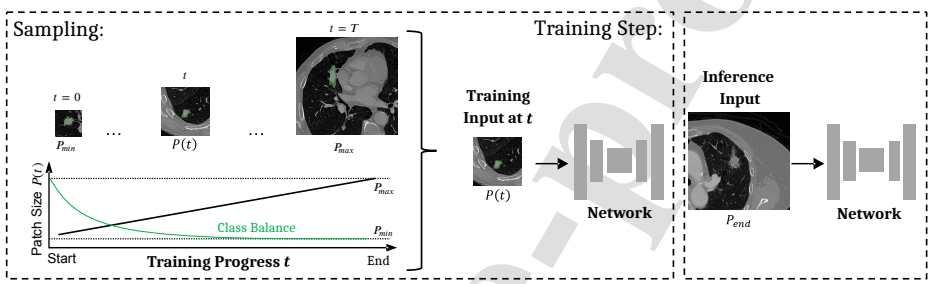
Please cite this article as: S.M. Fischer, J. Kiechle, L. Daza et al., Progressive growing of patch size: Curriculum learning for accelerated and improved medical image segmentation. *Medical Image Analysis* (2026), doi: https://doi.org/10.1016/j.media.2026.104195.

Graphical Abstract

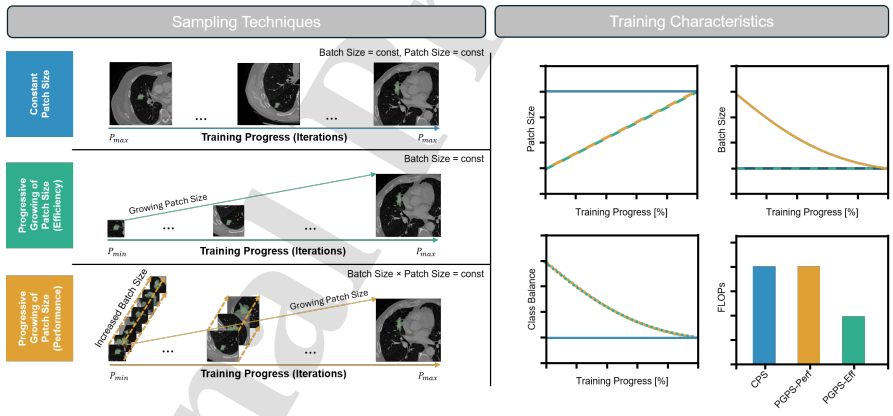
Progressive Growing of Patch Size: Curriculum Learning for Accelerated and Improved Medical Image Segmentation

Stefan M. Fischer, Johannes Kiechle, Laura Daza, Lina Felsner, Richard Osuala, Daniel M. Lang, Karim Lekadir, Jan C. Peeken, Julia A. Schnabel

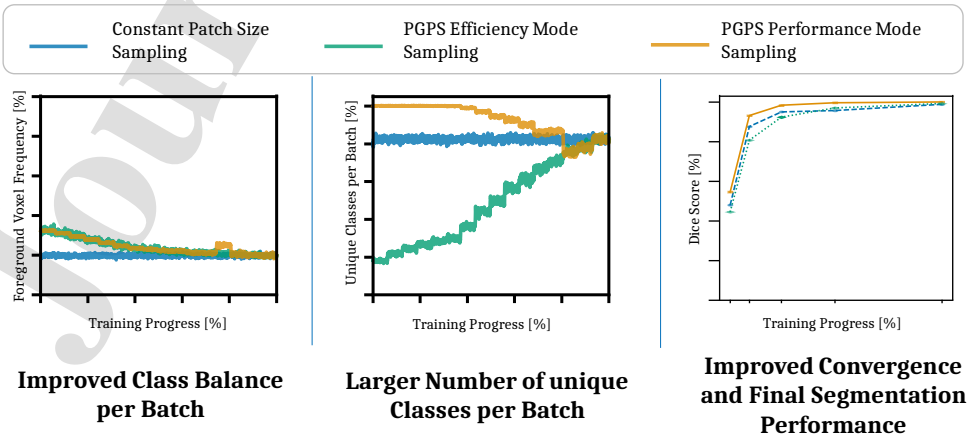
Advance 3D Patch-based Medical Image Segmentation Via Curriculum Learning



Apply Progressive Growing of Patch Size in Performance Mode or in Resource-Efficient Mode



Improved Training Dynamics via Progressive Growing of Patch Size over default Constant Patch Size Training



Improved Class Balance per Batch

Larger Number of unique Classes per Batch

Improved Convergence and Final Segmentation Performance

Highlights

Progressive Growing of Patch Size: Curriculum Learning for Accelerated and Improved Medical Image Segmentation

Stefan M. Fischer, Johannes Kiechle, Laura Daza, Lina Felsner, Richard Osuala, Daniel M. Lang, Karim Lekadir, Jan C. Peeken, Julia A. Schnabel

- Novel patch-size curriculum improves performance over fixed patch size training.
- Performance mode boosts Dice scores across all datasets with lower training cost.
- Efficiency mode matches Dice scores while cutting runtime and FLOPs sharply.
- Patch-size curriculum implicitly improves class balance for segmentation tasks.
- Simple to apply within standard patch-based 3D medical segmentation setup.

Progressive Growing of Patch Size: Curriculum Learning for Accelerated and Improved Medical Image Segmentation

Stefan M. Fischer[a,b,c,e,1], Johannes Kiechle[a,b,c,e], Laura Daza[a,c], Lina Felsner[a,c], Richard Osuala[a,c,g], Daniel M. Lang[a,c], Karim Lekadir[g], Jan C. Peeken[c,1,2], Julia A. Schnabel[a,c,e,f,2]

[a] *School of Computation, Information and Technology, Technical University Munich, Munich, Germany*
[b] *Department of Radiation Oncology, TUM School of Medicine, TUM University Hospital rechts der Isar, Technical University of Munich, Munich, Germany*
[c] *Institute of Machine Learning in Biomedical Imaging, Helmholtz Munich, Munich, Germany*
[d] *Institute of Radiation Medicine, Helmholtz Center Munich, Munich, Germany*
[e] *Munich Center of Machine Learning (MCML), Munich, Germany*
[f] *School of Biomedical Engineering and Imaging Sciences, King's College London, London, UK*
[g] *Departament de Matemàtiques i Informàtica, Barcelona Artificial Intelligence in Medicine Lab (BCN-AIM), Universitat de Barcelona, Barcelona, Spain*

Abstract

In this work, we introduce Progressive Growing of Patch Size (PGPS), an automatic curriculum learning approach for 3D medical image segmentation. Curriculum learning structures the training process by presenting progressively more complex samples to the model, often improving training convergence. In our case, we operationalize this by starting training with small patch sizes and gradually increasing them, which naturally improves the foreground-to-background class voxel ratio in early training stages. We evaluate our approach in two distinct settings. First, a resource-efficient mode maintains a constant batch size throughout training to reduce the input tensor size and computational cost (FLOPs) relative to conventional training. Second, a performance mode inversely scales the batch size relative to the patch volume, keeping the total FLOPs comparable to standard training while maximizing final segmentation quality. Both modes are evaluated on segmentation performance (Dice score) and computational costs across 15 diverse and popular 3D medical image segmentation tasks. The resource-efficient mode matches the segmentation performance of the conventional constant patch size baseline while reducing wall-clock training time to only 44%. We show that the performance mode improves upon the constant patch size baseline, achieving a statistically significant relative gain in mean Dice score of 1.28%. Remarkably, the performance mode surpasses the constant patch size baseline across all 15 tasks, while simultaneously reducing wall-clock training time to only 89%. We found that the benefits are particularly pronounced for tasks with severe foreground-to-background voxel imbalance, such as lesion segmentation. As a consequence of the improved convergence, the proposed performance mode reduces segmentation performance variance relative to conventional constant patch size training, making model comparisons less sensitive to training stochasticity. Finally, our experiments demonstrate that PGPS is not tied to a specific architecture but represents a broadly applicable strategy that consistently boosts performance across diverse segmentation models, including UNet, UNETR, and SwinUNETR. In summary, this simple yet effective transformation of the input sampling strategy substantially improves both segmentation performance and training efficiency, while remaining compatible with diverse segmentation backbones.

Keywords: 3D Medical Image Segmentation, Patch Sampling, Curriculum Learning, Class Imbalance

1. Introduction

Most research efforts in the area of medical image segmentation focus on the development of new architectural concepts, including convolution-based [1, 2], transformer-based [3, 4], and hybrid approaches [5, 6]. In contrast, comparatively little attention has been given to the training process itself, where models are still predominantly trained using random data sampling strategies. A growing body of work, however, suggests that the order in which samples are presented to a model matters.

Two prominent strategies embody this idea: active learning and curriculum learning. Both approaches require a scoring methodology to define a sample order that reflects, in some sense, the sample difficulty. Active learning [7, 8, 9, 10] operates at the data acquisition stage, prioritizing the annotation of samples expected to yield the greatest model improvement. Curriculum learning, by contrast, operates during model training itself: inspired by human learning, Bengio *et al.* [11] proposed ordering training samples from easy to hard, demonstrating that this structured progression leads to faster convergence and improved performance relative to random sampling. Curriculum learning also has the potential to substantially reduce the computational costs of training, both in terms of time and energy consumption, thereby benefiting the environment [12].

Several approaches have been proposed to design curricula for medical imaging. Some rely on human annotations or expert knowledge to define task-specific measures of diffi-

Email address: stefan.mi.fischer@tum.de (Stefan M. Fischer)
[1] Stefan M. Fischer is corresponding author
[2] Jan C. Peeken and Julia A. Schnabel contributed equally as senior authors.

culty [11, 13, 14]. For example, task difficulty might be linked to class membership, as in fracture classification [13], or defined using inter-rater agreement [13, 14]. However, expert annotations considerably increase the costs, making those approaches often infeasible. More general strategies estimate task difficulty automatically or synthetically change data complexity during training. Sample loss has been used as a proxy for difficulty, enabling hard-negative mining and adaptive oversampling, as shown in lung cancer segmentation [15]. In MRI-based brain tumor segmentation, Havaei *et al.* [16] have introduced a curriculum that improves robustness to missing modalities by randomly dropping channels with increasing probability during training. Karras *et al.* [17] have proposed to progressively grow a generative adversarial network layer by layer, effectively increasing task difficulty as output resolution doubled. This approach has been used for natural image generation and has shown improved convergence, reduced training time, and an overall better model performance. Zhao *et al.* [18] have extended this idea to semantic segmentation of cervical nuclei by progressively growing the UNet model from its bottleneck.

Another way to define task difficulty is by sample length. In natural language processing, curricula based on sentence length have been used to accelerate the training of large language models such as BERT and GPT-2 [19, 20, 21, 22] and have been adopted into the high-performance training library DeepSpeed [23]. In computer vision, an analogous measure is image size. Several works have employed curricula that (progressively) increase input resolution [24, 25, 26, 22, 27, 28, 29] during training. These methods consistently improve convergence, enabling either more resource-efficient training with comparable performance [24, 25, 27, 28, 29] or higher final accuracy within the same training budget [26]. Several curricula have been applied for visual backbone pretraining tasks [26, 25, 27, 28, 29], by pretraining on ImageNet classification [25, 24] or contrastive pretraining [26, 30, 31, 27, 28, 29]. In contrast, there is limited work in applying the sample-length curriculum directly to segmentation tasks [32], thus semantic segmentation mostly benefits from curriculum learning only indirectly via transferring pretrained weights [26, 25].

In this work, we propose a Progressive Growing of Patch Size curriculum, which is a novel sample-length curriculum for 3D medical image segmentation, defined through patch size. Our approach starts training on small patches, ensuring strong class balance in early training, and progressively increases to large patches, which provide a broader global context. Furthermore, the reduced memory footprint of smaller patches early in training allows for larger batch sizes under the same GPU budget. We argue that our proposed curriculum, built on patch size, is better suited for dense prediction tasks such as semantic segmentation than the Progressive Resolution curriculum applied in multiple methods in the CV domain [24, 25, 26]. Furthermore, our approach is an advancement over the conventionally applied constant patch size sampling.

We have introduced the key concept of Progressive Growing of Patch Size (PGPS) in our previous work at MICCAI 2024 [33], where we have established the fundamental idea of progressively increasing patch size during training. In this work, we significantly advance the initial concept by optimizing the methodology for computational efficiency and segmentation performance, and largely broadening the experimental and analytical scope in the following ways: (i) We propose a performance mode of the curriculum resulting in improved segmentation performance beyond the originally proposed method in [33], enabling cheap dataloading of very large batch sizes by efficiently sampling patches from a small number of volumes; (ii) We have investigated the relationship between dataset characteristics and the observed performance gains, identifying improved class balance as the key factor driving the enhanced results.; (iii) We have proven the generalization to different backbones by applying it successfully to one fully convolutional network, UNet, and two transformer-based networks, UNETR and SwinUNETR; (iv) We have analyzed the impact of stochastic training variability and show that the proposed performance mode notably reduces segmentation outcome variance compared to conventional constant patch size training; (v) We have conducted a direct comparison between our patch size–based curriculum and an existing resolution-based curriculum, showing that our approach consistently achieves superior segmentation performance.. In this work, we show that our proposed curriculum, built on patch size, improves segmentation performance in the form of Dice score, and at the same time reduces the computational costs of training compared to conventional constant patch size training.

2. Methods

This section describes the proposed methods and is split into four parts. As a basis, we describe conventional ways of sampling in patch-based medical segmentation. Then, we discuss the theoretical design of the proposed curriculum learning. After that, we introduce the efficient curriculum mode and the performance mode. Finally, we outline the nnU-Net framework implementation of the curricula.

2.1. Sampling in Patch-based 3D Medical Image Segmentation

Three-dimensional (3D) medical image segmentation is fundamentally constrained by the large GPU memory requirements of volumetric data processing. Common strategies to address this limitation include: (1) using low-resolution models [2], (2) applying high-resolution patch-based methods [6, 5, 34, 2, 35], and (3) employing network cascades [36] or multi-scale approaches [37]. Among these, it has been shown that high-resolution patch-based methods generally yield superior performance [36, 38]. Patch sampling strategies play a central role in patch-based methods. nnU-Net adopts a "forced oversampling" technique, where one foreground (FG) patch is sampled from a randomly chosen patient, with FG classes selected with equal probability, while another random patch is sampled from a different patient [2]. In contrast, MONAI implements "Probabilistic Oversampling" which applies probabilistic class sampling [39]. To maximize global context, standard practice is to use the largest possible patch size that fits into GPU memory, typically with a batch size of two [2]. Even with addi-

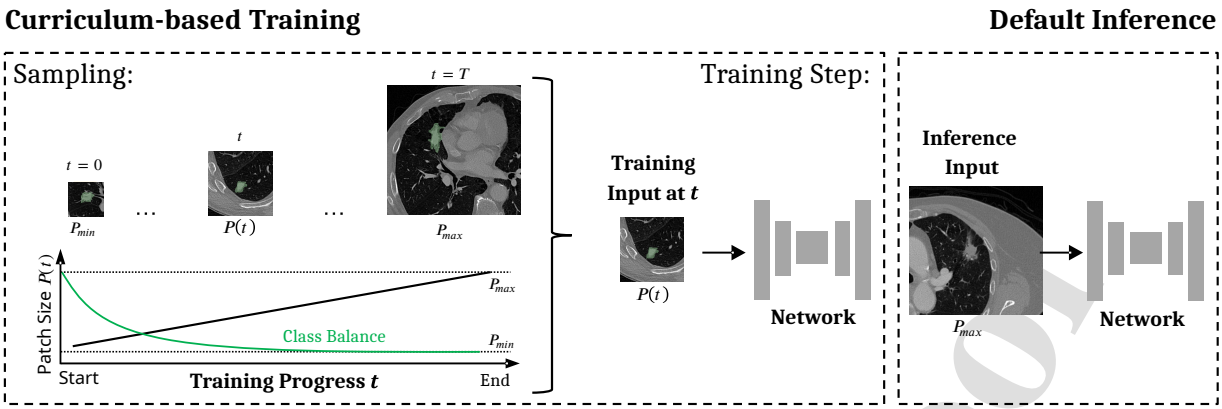


Figure 1: Overview of the proposed **Progressive Growing of Patch Size** curriculum, illustrated for lung cancer segmentation (cancer regions highlighted in green). Training begins with the minimal patch size P_{min} and progressively increases the patch dimensions until the final maximal patch size P_{max} is reached. Smaller patch sizes provide a better foreground-to-background voxel balance, which decreases as the patch size grows. During inference, the maximum patch size P_{max} is used to capture maximal global context. Figure is adapted from [33].

tional computational resources, nnU-Net prioritizes maximizing patch size over increasing batch size, as exemplified in its larger encoder variants such as nnU-Net ResEnc M/L/XL [40].

2.2. Progressive Growing of Patch Size

The central principle of our proposed PGPS curriculum is to begin the training with the smallest processable patch size and progressively increase to larger patches. The rationale for starting with smaller patches lies in task difficulty: smaller patches inherently yield a more balanced foreground-to-background voxel ratio, whereas this balance diminishes as patch size increases. In the theoretical case of a single-voxel patch, a batch containing one foreground and one background patch would achieve perfect class balance, with half of the voxels belonging to the foreground and half to the background. As the patch size grows, however, the distribution converges toward the full volume statistics, which are typically dominated by background.

Patch size increments are applied in the smallest feasible steps, with each axis adjusted independently. By gradually increasing the patch size in minimal increments, transitions between training stages are smoothed, as the segmentation tasks at successive patch sizes remain closely related. This design provides the network with a structured sequence of progressively more challenging tasks. An overview of this curriculum is illustrated in Figure 1.

Both the minimal patch size and the patch size increment depend on the input size constraints of the underlying network architecture. The progression continues until the largest possible patch size is reached, which is typically constrained by the GPU memory capacity or set to the network's default patch size. During inference, the largest patch size is employed to maximize global context, which usually achieves the highest segmentation performance [2].

2.3. Curriculum Modes: Efficiency vs. Performance

We propose two modes of the PGPS curriculum, tailored to different objectives. **PGPS-Efficiency** minimizes training runtime while maintaining performance comparable to standard constant patch size sampling. **PGPS-Performance** aims to maximize segmentation performance. Intermediate configurations between these two extremes are also possible. Figure 2 illustrates both modes for a lung lesion segmentation task.

PGPS-Efficiency: In PGPS-Efficiency, the batch size is kept constant throughout training. Since the patch size is smaller at early stages, the number of computations is reduced, leading to a substantial reduction in training time. This is illustrated in Figure 2 in green.

PGPS-Performance: In PGPS-Performance, the GPU memory budget is fully utilized by dynamically increasing the batch size when smaller patches are used. This design leverages available resources more effectively, with the goal of achieving the highest possible segmentation performance. This is illustrated in Figure 2 in yellow.

2.4. nnU-Net Framework Implementation

In the following section, we describe the integration details of default constant patch size sampling and the two different proposed PGPS sampling modes into the nnU-Net framework [2], which is the state-of-the-art 3D medical image segmentation framework [40]. A detailed pseudo code for both curriculum modes implemented into the nnU-Net framework is given in Algorithm 1.

2.4.1. Baseline: Constant Patch Size Sampling

The baseline sampling method utilized in this study is fixed or constant patch size (CPS) sampling. This sampling strategy corresponds to the conventional configuration of training a (patch-based) segmentation network.

We have chosen nnU-Net [2] as the underlying framework, as it provides state-of-the-art performance and is auto-configuring to downstream tasks. The default nnU-Net configuration is designed to achieve maximum global context by employing the largest possible patch size that fits within nnU-Net's typical GPU budget of 8GB.

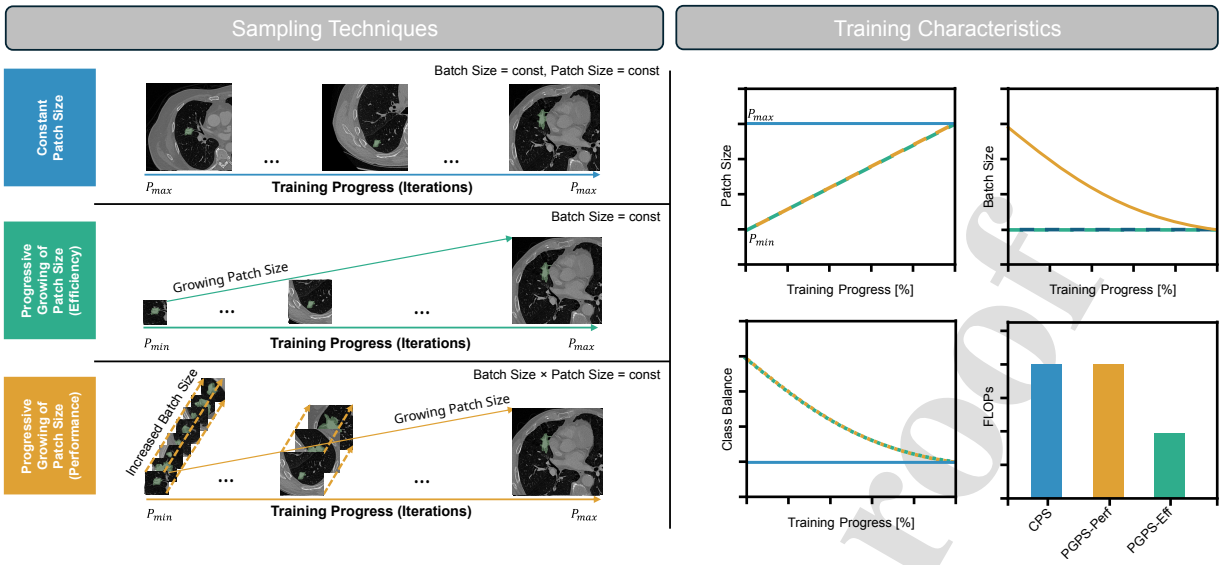


Figure 2: Comparison between vanilla constant patch size (CPS) training and the two proposed modes of the **Progressive Growing of Patch Size** (PGPS) curriculum for lung lesion segmentation. In both curriculum modes, the patch size is progressively increased during training. In PGPS-Efficiency (green), the batch size remains constant, resulting on average in smaller input tensors. In PGPS-Performance (orange), the available GPU budget is fully utilized by inversely scaling the batch size to the patch volume. On the right side, the general training characteristics for all three sampling techniques are plotted. Here, the patch size follows a simple linear function for the PGPS modes. PGPS results in increased foreground-to-background voxel balance. While FLOPs for CPS and PGPS-Performance are comparable, PGPS-Efficiency significantly reduces computational costs.

Given a patch-based setting, where the maximal processable patch size is smaller than the target volume size, nnU-Net will by default apply a batch size of two. In this setting, two patients are randomly sampled, while from the first patient, a foreground (FG) patch will be created, and from the second patient, a random patch will be sampled. This default 50% FG patch ratio for a batch is designed to improve robustness in the presence of class imbalance [2]. If multiple FG classes are present, one class will be picked randomly with equal probabilities. If nnU-Net sets a batch size larger than two, the framework employs a FG patch ratio of 33%.

2.4.2. PGPS Curricula

We integrate the proposed PGPS curricula within the nnU-Net framework. This integration involves maintaining the default nnU-Net settings while only adjusting the patch size for PGPS-Efficiency and PGPS-Performance, while also increasing the batch size for PGPS-Performance.

For the PGPS curricula, we have to define the minimal processable patch size and, given the maximal patch size, find all processable intermediate patch sizes that the network can process. In the case of nnU-Net, which applies a fully convolutional UNet backbone, the smallest processable patch size is dependent on the number of downsampling operations. The minimum patch size length depends on the number of downsampling operations for a UNet, and is given by 2^{num_pool}, where *num_pool* is the number of pooling operations for each axis. All intermediate patch sizes have to be fully divisible by 2^{num_pool}. The maximal patch size is set to the patch size nnU-Net would by default apply to the given task.

The patch size is increased in a step-wise linear fashion from the minimal possible patch size that the UNet allows until the patch size reaches the nnU-Net's target patch size. To have a smaller input tensor size growth, we only increase the patch size for one axis per step. We always select the axis with the lowest numerical value to increase the patch size. This scheme results in a higher average batch size during training, compared to naively sequentially increasing each axis. We train each patch size stage with the same number of epochs. Implementation differences between the newly proposed PGPS modes in this article and the original MICCAI publication modes are given in detail in Appendix E.

PGPS-Efficiency: For PGPS-Efficiency, only the patch size is adjusted during training, while the batch size is kept constant, using the default nnU-Net batch size. This results in reduced GPU memory utilization for smaller patch sizes.

PGPS-Performance: In PGPS-Performance, both patch size and batch size are adjusted to fully utilize the available GPU memory budget. By default, nnU-Net enforces a foreground patch ratio of 33% for batch sizes larger than two, but switches to 50% when the batch size is two. This setting would introduce a discontinuity in the class balance trajectory. To ensure a smooth progression, we instead fix the foreground patch ratio to 50%, regardless of batch size. As there are different options in how to increase the batch size, we evaluated multiple batch construction strategies regarding efficiency and segmentation performance in Appendix A. Loading large numbers of volumes can increase training wall-clock time significantly as the dataloading becomes the training bottleneck. As a result, we explored whether creating multiple patches per volume is an effective strategy to overcome this limitation. We found that creating all patches from two patients per batch, using one pa-

tient for foreground patches and the other for random patches, achieves the best efficiency and segmentation performance. We follow this batching strategy throughout the manuscript.

3. Experiments

In Section 3.1, we introduce a set of widely used public medical image segmentation datasets used for the experiments. In Section 3.2, we evaluate the PGPS curriculum modes, analyze segmentation performance, computational costs (runtime and FLOPs), and convergence properties, and compare them to the constant patch size baseline. We present a correlation analysis between performance differences and task-specific characteristics in Section 3.3. A comparison between Progressive Resolution [24] and our proposed method is provided in Section 3.4. In Section 3.5, the generalization of PGPS to different vision backbones is investigated. Finally, the stochastic training variability of the model performance across repeated trainings within the three sampling strategies is examined in Section 3.6.

3.1. Dataset Descriptions

To test the curriculum methods, a wide range of publicly available segmentation datasets is utilized. In total, 15 different 3D medical semantic segmentation datasets have been gathered. The dataset mix includes the Medical Segmentation Decathlon [41], comprising 10 distinct datasets covering lesion, organ, and vessel tasks. Additionally, the ToothFairy2 dataset [42], TotalSegmentatorV2 [43], KiTS23 [44], AMOS22 CT/MRI task [45] and BTCV [46] are included. These datasets encompass a wide range of segmentation tasks, including lesions, various organs, bones, muscles, teeth, implants, and nerves. Furthermore, the datasets represent different 3D modalities such as CT, Conebeam-CT, and MRI, and vary in size from small to large, ranging from 20 to 1200 samples. They present a diverse range of class numbers, from 2 (binary) to 117 classes.

3.2. Benchmarking of Methods on 15 Datasets

We evaluate segmentation performance, training runtime, and computational cost (FLOPs) of the PGPS curricula relative to default constant patch size (CPS) training. The default nnU-Net preprocessing and training pipeline is applied, using the 3D high-resolution patch-based variant for all tasks. For ToothFairy2 and TotalSegmentatorV2, mirroring data augmentation is disabled, as prior studies have demonstrated improved results without mirroring [47, 43]. For both datasets, we also evaluate the segmentation performance for nnU-Net training without mirroring data augmentation in Appendix B.

3.2.1. Segmentation Performance

To benchmark CPS, PGPS-Efficiency, and PGPS-Performance, we follow the standard nnU-Net evaluation procedure: models are trained using 5-fold cross-validation, and Dice scores are averaged across all foreground classes.

To assess whether segmentation performance differs significantly between CPS and the PGPS curricula across the 15 datasets, paired Wilcoxon signed-rank tests are performed on the average foreground Dice scores. For this, we built one sample per dataset by pairing curriculum and CPS Dice score.

3.2.2. Training Runtime Tracking

Training time is monitored, excluding validation. Since experiments have been conducted on a cluster with heterogeneous GPU and CPU configurations, we propose a virtual relative runtime to enable fair comparisons. For the PGPS curricula, time per epoch was recorded. We then compute the average runtime for all maximal patch size epochs, which then represents the runtime of a single CPS epoch. Given that, we can then compute the virtual runtime of a full CPS training under the same hardware settings. We report the median across all 5 folds.

3.2.3. FLOPs Tracking

The number of Floating Point Operations (FLOPs) is directly proportional to the size of input tensors. We recorded the total number of iterated voxels during training for each strategy and normalized them to the CPS value, yielding the relative difference in computational cost between PGPS curricula and CPS.

3.2.4. Convergence Analysis

We analyze the convergence behavior of the three sampling strategies. For each strategy, models are trained across all 15 datasets with varying training lengths: 1%, 10%, 25%, 50%, and 100% of total training iterations. This is done by reducing the number of iterations per epoch relative to the default 250 training iterations for nnU-Net. Evaluation at each training length is performed using 5-fold cross-validation.

3.3. Task Characteristics Analysis

To identify dataset characteristics associated with segmentation performance improvements, we correlate relative performance gains with the following task measures: (1) number of semantic classes, (2) training dataset size, (3) patch-to-volume coverage (the proportion of a full volume seen in a single patch), and (4) class imbalance, measured by the frequency of the smallest class.

Spearman correlation tests are conducted for each task characteristic against the relative Dice score improvement between CPS and PGPS curricula. Separate tests are performed for each training length (1%, 10%, 25%, 50%, and 100%).

Additionally, we record the foreground ratio of input tensors, the number of unique classes seen per iteration, and the patch size ratio across all three sampling strategies.

3.4. Progressive Resolution Curriculum

Another input-length curriculum in computer vision is progressive resizing, also called Progressive Resolution [26, 24], which gradually increases input resolution from low to high. This approach also modifies input length, transitioning from small to large input tensors. To compare our PGPS curricula with Progressive Resolution, we evaluated both on BTCV, AMOS22, KiTS23, and MSD Lung Cancer, covering two highly imbalanced lesion tasks and two multi-organ tasks.

Table 1: Performance of CPS, PGPS-Efficiency, and PGPS-Performance on 15 diverse 3D Medical Image Segmentation tasks. CPS refers to constant patch size training, which is the standard nnU-Net training. All models have been trained from scratch. **Dice score [%]:** evaluated in 5-fold cross-validation as in [36]; ▲*x*% **Rel. Dice score**: Relative Dice score differences to CPS (▲ increase, ▼ decrease). **Rel. Runtime [%]:** relative runtime normalized to CPS's runtime; **Rel. FLOPs [%]:** relative count of floating point operations to CPS value. **Bold**: Best performing sampling. Underlined: Second best performing sampling.

Dataset	Dice score [%] ↑			▲*x*% Rel. Dice score ↑		Rel. Runtime [%] ↓		Rel. FLOPs [%] ↓	
	CPS	PGPS-Eff	PGPS-Perf	PGPS-Eff	PGPS-Perf	PGPS-Eff	PGPS-Perf	PGPS-Eff	PGPS-Perf
MSD Brain	74.12	**74.29**	74.15	▲ 0.23	▲ 0.04	42	83	34	84
MSD Heart	93.29	93.24	**93.29**	▼ 0.07	▲ 0.00	40	88	31	89
MSD Liver	78.74	78.76	**80.60**	▲ 0.03	▲ 2.36	51	85	38	84
MSD Hippocampus	88.96	89.12	**89.15**	▲ 0.20	▲ 0.22	67	112	33	92
MSD Prostate	73.13	75.26	**76.31**	▲ 2.91	▲ 4.35	43	87	30	87
MSD Lung	70.00	72.30	**72.77**	▲ 3.29	▲ 3.96	41	82	31	89
MSD Pancreas	68.68	68.60	**68.80**	▼ 0.12	▲ 0.17	39	87	31	88
MSD Hepatic Vessel	68.61	68.05	**68.98**	▼ 0.82	▲ 0.54	47	87	33	88
MSD Spleen	97.02	95.85	**97.15**	▼ 1.21	▲ 0.13	42	86	33	89
MSD Colon	48.41	50.83	**51.02**	▲ 5.00	▲ 5.39	38	87	28	90
BTCV	83.37	83.05	**83.81**	▼ 0.38	▲ 0.53	41	88	29	89
KiTS23	86.02	**87.22**	86.46	▲ 1.40	▲ 0.51	45	86	27	84
AMOS22	88.62	88.10	**88.78**	▼ 0.59	▲ 0.18	49	87	33	89
ToothFairy2	76.92	**77.15**	77.00	▲ 0.31	▲ 0.11	34	85	26	88
TotalSegmentatorV2	87.82	85.09	**88.15**	▼ 3.11	▲ 0.38	33	108	30	90
Norm. Avg.	100.00	100.47	**101.26**	▲ 0.47	▲ 1.26	44 ± 9.5	89 ± 8.7	33 ± 2.6	88 ± 2.4

We implement the Progressive Resolution curriculum within the nnU-Net framework as follows: Input tensor sizes are matched to those used for the PGPS curricula; however, instead of adjusting crop sizes, the default nnU-Net patch size is resampled to fit PGPS' target tensor size for each phase. Increasing the batch size, as done in PGPS-Performance, is computationally to expensive within nnU-Net for Progressive Resolution; therefore, we only evaluate Progressive Resolution using a fixed batch size. Segmentation performance of both curricula is compared using Dice scores in a 5-fold cross-validation.

3.5. Different Architectures

The proposed curriculum can be applied to any vision backbone that supports flexible input sizes. To assess its applicability beyond CNNs, we further evaluated the curriculum on transformer-based hybrid architectures, specifically UNETR [5] and SwinUNETR [6], both combining a transformer encoder with a CNN decoder.

In transformer-based models, architectural constraints define the minimal feasible patch size and increment: for UNETR, this is determined by the internal patchify size (distinct from the patch size used in patch-based training), while for SwinUNETR it is defined by the Swin window size. As transformers typically rely on fixed input dimensions due to positional embeddings, we interpolate the positional embeddings for UNETR to match the varying patch sizes as in [3]. SwinUNETR does by default not use any positional embeddings.

We integrate both architectures into the nnU-Net framework, train following the standard nnU-Net training, and evaluate on the BTCV dataset using a 5-fold cross-validation. All backbones are trained from scratch.

3.6. Stochastic Training Variability

Neural network training is inherently stochastic, a property that is amplified in patch-based pipelines, where both the sampled volume and patch location are randomly selected (within oversampling constraints). Consequently, repeated training runs can yield different models and segmentation performance.

To quantify this variability, we repeat training on a single fold of the 5-fold cross-validation five times for each strategy. Experiments are conducted on the BTCV multi-organ segmentation dataset and the highly class-imbalanced MSD Lung Tumor dataset. Training lengths of 1%, 10%, and 100% of the default iteration count were analyzed.

To assess the stochastic impact of sampling, we build triplets of each strategy's segmentation performance outcome and evaluate all 125 possible outcome combinations per training length. As each single outcome could be a potential result, we are interested in which sampling strategy ranks best in each possible comparison. Thus, we count the number of scenarios in which each strategy outperformed both other strategies.

4. Results

4.1. Benchmarking of Methods on 15 Datasets

We evaluate the PGPS curricula relative to the conventional constant patch size (CPS) baseline across 15 popular 3D medical image segmentation tasks. Key measures include segmentation performance, training runtime, and computational cost in terms of FLOPs. Additionally, we also analyze the training convergence.

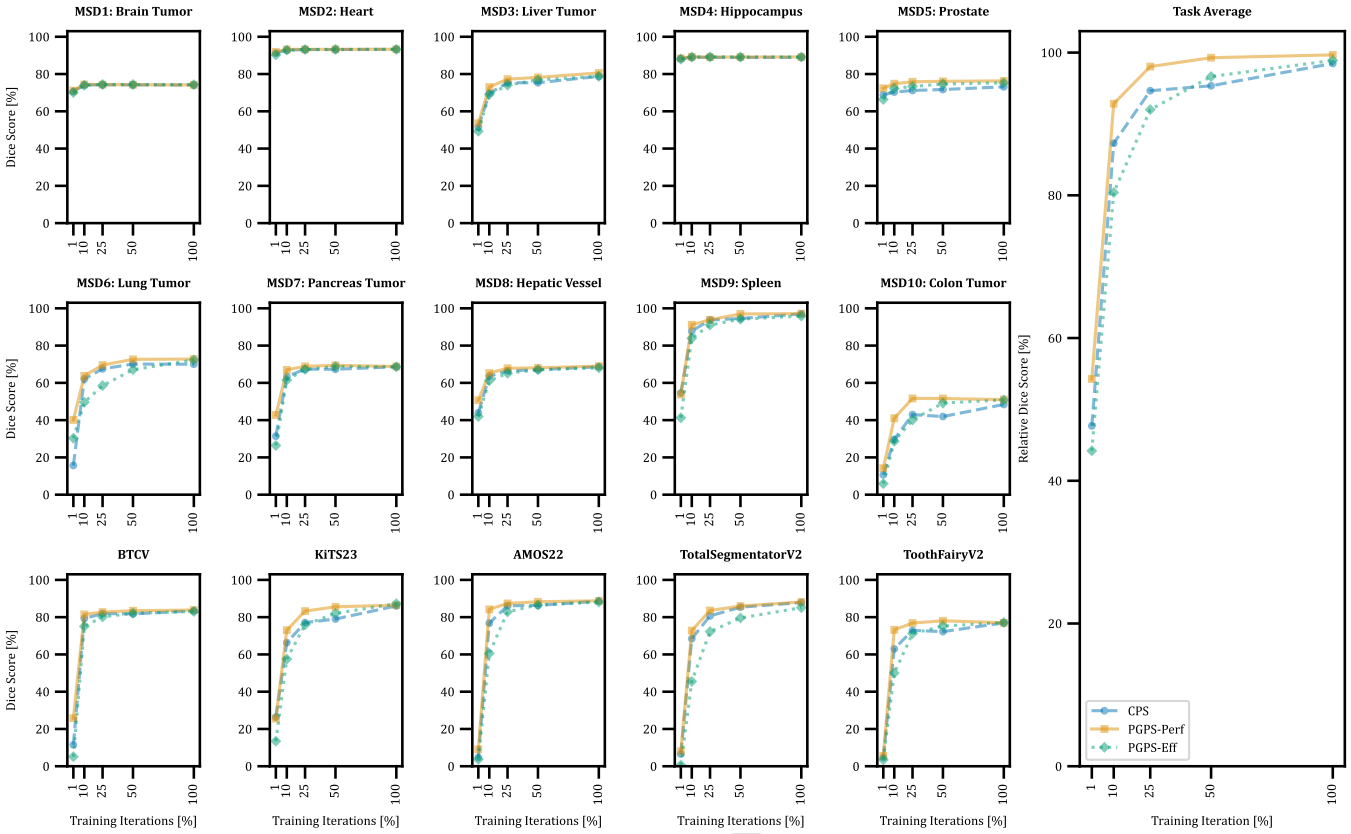


Figure 3: Segmentation performance convergence of CPS, PGPS-Performance, and PGPS-Efficiency across training iterations. Dice scores are tracked across 15 segmentation tasks for different training lengths (1%, 10%, 25%, 50%, and 100% of nnU-Net's default total training iterations). On average, PGPS-Performance exhibits the fastest convergence, while PGPS-Efficiency converges more slowly, due to the smaller input tensors that result in superior training speed. The final performance is, on average, best for PGPS-Performance, followed by PGPS-Efficiency and CPS. The average convergence over all 15 datasets (right) is computed by normalizing each task by its maximum performance and then averaging across all tasks.

4.1.1. *Segmentation Performance*

Table 1 summarizes the benchmarking between baseline and the curricula approaches on 15 datasets. Across the 15 datasets, PGPS-Performance achieves the highest average Dice score, yielding a relative improvement of 1.26% over CPS. It outperforms CPS in every task, while ranking as the best-performing strategy in 12 of the 15 tasks. In all 15 datasets, PGPS-Performance is among the top two strategies, and only in the three tasks, KiTS23, ToothFairy2, and MSD Brain Tumor, PGPS-Performance is surpassed by PGPS-Efficiency. PGPS-Efficiency also delivers modest overall performance gains, with a relative improvement of 0.47% over all 15 tasks compared to CPS. It outperforms CPS in 8 tasks and is leading in 3 tasks. However, in 7 tasks, it has the lowest Dice score. A two-sided paired Wilcoxon signed-rank test comparing PGPS-Performance to CPS across all 15 datasets reveals a significant improvement for PGPS-Performance ($p \approx 0.0001$). Additionally, PGPS-Performance significantly outperforms PGPS-Efficiency ($p \approx 0.0103$), confirming it as the superior strategy for segmentation performance. A two-sided paired Wilcoxon signed-rank test between PGPS-Efficiency and CPS shows no significant difference ($p \approx 0.6788$), indicating comparable segmentation performance between these two strategies.

4.1.2. *Training Runtime Tracking*

Table 1 shows the relative runtime for all 15 datasets. PGPS-Performance requires approximately 89% of CPS's runtime, ranging from 82% to 112% across tasks. Tasks for which PGPS-Performance yields longer training times than CPS, i.e. MSD Hippocampus and TotalSegmentatorV2, have the two largest start batch sizes of 1575 and 784, respectively. In total, the 15 tasks have an average batch size of around 399, ranging from 128 to 1575. PGPS-Efficiency substantially reduced training time, requiring only about 44% of CPS runtime (range: 34%–67%), achieving the fastest training while maintaining comparable segmentation performance.

4.1.3. *FLOPs Tracking*

As shown in Table 1, PGPS-Efficiency utilizes only about 33% of the FLOPs of CPS on average, with a range of 26%–38%. Although PGPS-Performance is designed to maximize the used GPU memory, it requires slightly fewer FLOPs than CPS, averaging 88% (range: 84%–92%). The reduced computational cost arises from the discrete nature of batch and patch sizes, which prevents full usage of the GPU budget in every epoch, resulting in lower FLOPs.

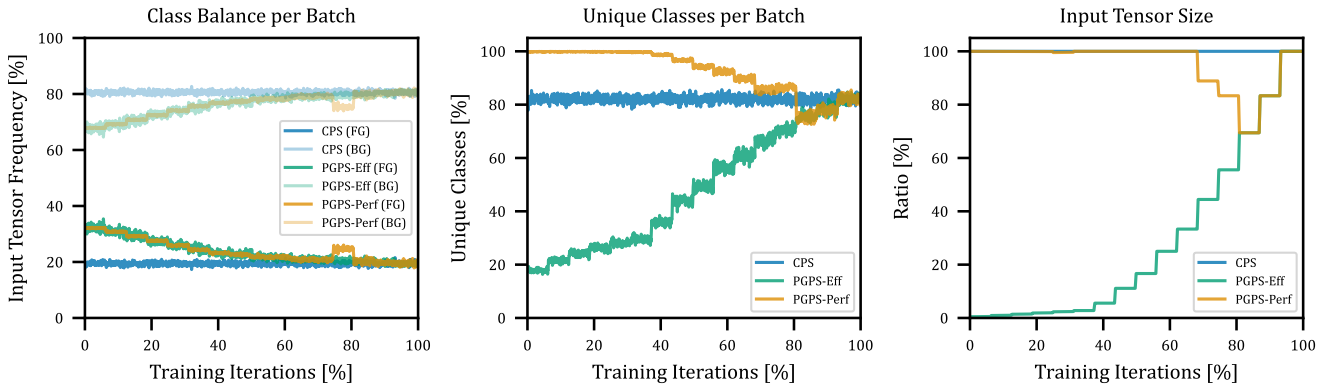


Figure 4: Training characteristics of PGPS curricula for the BTCV dataset with 14 classes. **Left**: PGPS curricula improve foreground-background voxel balance compared to CPS due to on average smaller patch sizes. **Center**: PGPS-Performance achieves the highest average number of unique classes per batch/iteration due to larger batch sizes, while PGPS-Efficiency has the lowest. **Right**: CPS maintains a constant input tensor size; PGPS-Efficiency increases tensor size exponentially, while PGPS-Performance exhibits occasional size drops due to the discrete nature of patch and batch sizes.

4.1.4. Convergence Analysis

Figure 3 shows the convergence of segmentation performance across all 15 tasks over the number of iterations.

At 1% training length, PGPS-Performance already achieves the highest Dice score in 13 tasks, ranking second to CPS only in MSD Spleen and KiTS23. At 10% and 25%, PGPS-Performance outperformed both other strategies across every task. At 50%, PGPS-Performance outperforms both other strategies except for MSD Heart. However, the minimal differences in the results across methods and between training at 50% and 100% suggest that the performance is saturated in the MSD Heart task.

Overall, PGPS-Performance consistently converges faster than both CPS and PGPS-Efficiency, achieving superior results in the majority of training-length scenarios for each task.

In contrast, PGPS-Efficiency exhibits slower convergence than both other sampling strategies, as shown in Figure 3, but requires substantially fewer computations, as seen in Table 1, processing only about one-third of the voxels. Nevertheless, over the course of full training, PGPS-Efficiency reaches comparable, and in some cases even superior segmentation performance compared to CPS.

4.2. Task Characteristics Analysis

To better understand the improved convergence for PGPS curricula, we track several task characteristics during model training for the BTCV dataset. Tracked characteristics include foreground class balance, number of unique classes per iteration, and patch size ratio. These are illustrated in Figure 4. We find that both PGPS curricula exhibit higher class balance during early training phases compared to CPS, which maintains at a stationary foreground-to-background ratio. The reason is that fewer surrounding background voxels are contained in smaller foreground patches. At the final stage, PGPS curricula converge to the same class balance as CPS due to identical patch sizes. The standard deviation of class balance is larger for PGPS-Efficiency than for PGPS-Performance. PGPS-Performance shows a discontinuous drop in class balance during later training stages, attributed to the discrete nature of batch and patch sizes.

Tracking the number of unique classes per iteration for the 14-class BTCV task, PGPS-Performance produces batches containing nearly all semantic classes ($\sim$ 100%) during early patch size stages, converging to the stationary CPS value ($\sim$ 82%) in later stages. Small drops in unique classes per iteration are again observed due to the discrete batch and patch sizes. PGPS-Efficiency begins with the smallest fraction of unique classes per iteration ($\sim$ 18%), which gradually increases across patch size stages, eventually converging to the CPS value. The input tensor sizes of PGPS-Performance are the same as for CPS, but have occasional drops due to the discrete nature of batch and patch size. These drops result in the shorter training runtime observed for PGPS-Performance relative to CPS. PGPS-Efficiency exhibits a monotonic, stepwise exponential increase in patch size ratio.

PGPS-Performance: Spearman tests do not yield any significant correlation between PGPS-Performance relative improvements and the dataset characteristics of the number of semantic classes, dataset size, or patch-to-volume coverage. The only significant correlation is negative, occurring for class imbalance measured via the smallest class frequency at training lengths of 1% ($\rho \approx -0.7429$, $p \approx 0.0015$) and 10% ($\rho \approx -0.5464$, $p \approx 0.0351$). This indicates that PGPS-Performance is particularly beneficial for highly imbalanced tasks.

PGPS-Efficiency: A significant positive correlation is observed between class imbalance (smallest class frequency) and relative performance at training lengths of 1% ($\rho \approx 0.5321$, $p \approx 0.0412$), 10% ($\rho \approx 0.7679$, $p \approx 0.0008$), and 25% ($\rho \approx 0.6071$, $p \approx 0.0134$). Highly imbalanced tasks show reduced performance in short training scenarios, but PGPS-Efficiency eventually outperforms CPS at full training. This trend is evident for lesion tasks such as MSD Brain, MSD Liver Tumor, MSD Prostate, MSD Lung Tumor, MSD Colon, and KiTS23. Addi-

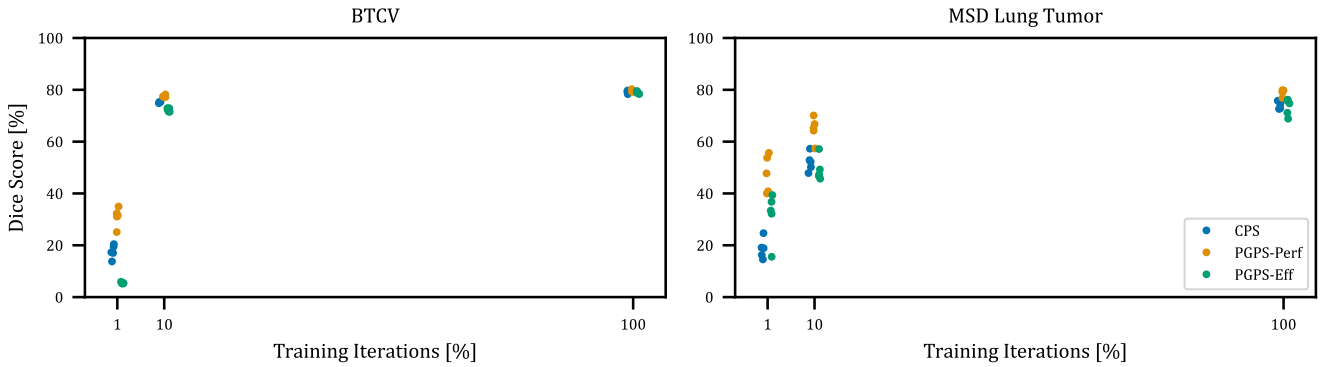


Figure 5: Segmentation performance for repeated training of CPS, PGPS-Efficiency, and PGPS-Performance across different training lengths (1%, 10%, and 100% of nnU-Net's total training iterations). Each experiment is repeated five times on the same data split with different random seeds. Variability decreases with longer training, while mean performance increases. The highly class-imbalanced MSD Lung Tumor dataset exhibits higher variance than the multi-class BTCV dataset. PGPS-Performance shows improved convergence and less performance variability between repeated runs. PGPS-Efficiency results in slower convergence than CPS and PGPS-Performance and has the highest performance variability. PGPS-Performance outperforms both other strategies in 100% of the 125 possible combinations for MSD Lung Tumor, while for BTCV, PGPS-Performance outperforms both other strategies in 58.4% of all combinations, as well as CPS and 60% of combinations.

tionally, patch-to-volume coverage correlates significantly with relative performance at training lengths of 10% ($\rho \approx 0.5157$, $p \approx 0.0491$) and 25% ($\rho \approx 0.5318$, $p \approx 0.0382$).

4.3. Progressive Resolution Curriculum

We evaluate the Progressive Resolution curriculum, an input-length strategy widely applied in computer vision [24, 25, 26], by adapting it to 3D patch-based medical image segmentation. Table 2 shows Dice scores for Progressive Resolution and PGPS-Efficiency. We find that our proposed PGPS-Efficiency consistently outperforms Progressive Resolution.

In the lesion segmentation task MSD Lung Tumor, Progressive Resolution lags behind PGPS-Efficiency by 2.9 Dice points (69.42% vs. 72.30%), and in KiTS23 by 1.2 points (84.63% vs. 87.22%). These results highlight that Progressive Resolution struggles in highly imbalanced settings, where PGPS-Efficiency provides a clear advantage.

For multi-organ segmentation tasks, such as BTCV and AMOS22, the performance gap is smaller. Progressive Resolution achieves 82.93% Dice on BTCV, close to PGPS-Efficiency (83.05%). Similarly, in AMOS22, it reaches 87.98%, compared to 88.10% for PGPS-Efficiency. Although differences in these multi-organ tasks are less pronounced, Progressive Resolution is still the lowest-performing strategy for all cases.

Table 2: Dice scores of Progressive Resolution (Prog. Res.) and Progressive Growing of Patch Size (PGPS) curricula. PGPS increases patch size during training, whereas Progressive Resolution increases patch resolution. PGPS-Efficiency consistently outperforms Progressive Resolution, with the largest gains observed in highly imbalanced lesion tasks. Across all datasets, Progressive Resolution is the lowest-performing strategy. Results are reported as mean Dice score (%) of a 5-fold Cross-Validation. **Bold:** best-performing strategy.

Dataset	Prog. Res.	PGPS-Eff
MSD Lung Tumor	69.42	**72.30**
KiTS23	84.63	**87.22**
BTCV	82.93	**83.05**
AMOS22	87.98	**88.10**

4.4. Different Architectures

We further assess the generalizability of PGPS by applying it to transformer-based backbones (UNETR, SwinUNETR). Results for the BTCV dataset are reported in Table 3 and compared to nnU-Net's default UNet backbone.

PGPS-Performance consistently improves segmentation performance across all architectures. Compared to CPS, Dice scores increase by +0.44 points for nnU-Net's fully convolutional UNet (83.37% → 83.81%), +1.56 points for UNETR (71.20% → 72.76%), and +3.77 points for SwinUNETR (71.62% → 75.39%). These results indicate that PGPS-Performance provides benefits not only for convolutional but also for transformer-based models.

In contrast, PGPS-Efficiency yields mixed results. For nnU-Net, performance slightly decreases by −0.32 Dice points (83.37% → 83.05%), and for SwinUNETR it improves marginally by +0.36 points (71.62% → 71.98%). For UNETR,

Table 3: Impact of PGPS-Efficiency (PGPS-Eff) and PGPS-Performance (PGPS-Perf) on different backbones for the BTCV task. Results are reported as mean Dice score (%) across five folds. PGPS-Performance consistently outperforms CPS across all architectures. PGPS-Efficiency converges for UNet and SwinUNETR but fails for UNETR due to gradient instability. Results are reported as mean Dice score (%) of a 5-fold Cross-Validation. **Bold:** best-performing strategy. Underlined: second-best-performing strategy.

Backbone	CPS	PGPS-Eff	PGPS-Perf
UNet (nnU-Net) [2]	83.37	83.05	**83.81**
UNETR [5]	71.20	0.00	**72.76**
SwinUNETR [6]	71.62	71.98	**75.39**

however, all training runs with PGPS-Efficiency diverge due to gradient explosion.

4.5. Stochastic Training Variability

Figure 5 summarizes the variability in segmentation performance under repeated training. MSD Lung Tumor shows overall higher segmentation performance variability than for the BTCV dataset. Two consistent trends are observed: (i) outcome deviation per strategy decreases as training length increases, and (ii) sampling strategy PGPS-Performance dominates across both datasets and training length scenarios in segmentation performance over CPS and PGPS-Efficiency.

BTCV: PGPS-Performance dominates early training (1–10%), achieving the highest segmentation performance in all comparisons against CPS and PGPS-Efficiency. At full training, it still achieves the majority of wins (58.4%) over CPS and PGPS-Efficiency and the highest Dice score ($79.57\% \pm 0.37$), followed by CPS ($79.16\% \pm 0.47$) and PGPS-Efficiency ($78.85\% \pm 0.41$).

MSD Lung Tumor: Across all training lengths, PGPS-Performance wins every sampling strategy comparison and achieves the highest Dice score ($78.98\% \pm 1.10$), outperforming CPS ($74.11\% \pm 1.23$) and PGPS-Efficiency ($73.34\% \pm 2.86$). Variability in segmentation performance per sampling strategy is larger at short training lengths but stabilizes as training progresses.

5. Discussion

In this work, we introduced a novel curriculum learning strategy that progressively increases patch size during training. The curriculum was implemented in two modes: one optimized for segmentation performance and the other for computational efficiency. We have evaluated Dice score performance, training convergence, and computational cost across 15 diverse datasets covering lesion, multi-organ, vessel, muscle, and skeletal segmentation, and compared it to standard constant patch size (CPS) sampling. The performance mode demonstrates substantial Dice score improvements over CPS while simultaneously reducing computational costs in terms of FLOPs and training time. The efficiency mode matches the segmentation performance of CPS, while drastically cutting both FLOPs and training time by more than half. In addition, we have successfully applied the curriculum on multiple network backbones, including UNet, UNETR, and SwinUNETR, thereby confirming its broad applicability. In the following, we discuss the key factors driving these performance gains, as well as the limitations and implications of the Progressive Growing of Patch Size (PGPS) approach. We added a detailed comparison between the original proposed PGPS modes in [33] and the newly proposed modes in Appendix E. The newly proposed modes outperform the original modes in segmentation performance and computational efficiency.

5.1. Handling Class Imbalance

Our proposed curriculum is a novel approach for tackling class imbalance in image segmentation. PGPS implicitly enhances class balance, as smaller patches contain proportionally fewer background voxels. This effect is particularly beneficial for imbalanced tasks, as class balance has shown a significant correlation with improved segmentation performance in our experiments.

Another approach for improving class balance is Curriculum Adaptive Sampling for Extreme Data Imbalance (CASED) [15], which explicitly increases foreground oversampling at the beginning of training and gradually reduces it over time. While CASED relies on a large batch size, it trades off patch size (e.g., patch size of 68^3 and batch size of 16). Our method, however, follows the configuration principles of modern segmentation frameworks such as nnU-Net, which prioritize a large patch size over the batch size (typically batch size of 2). In principle, CASED could be integrated into PGPS to further enhance class balance, offering a promising direction for future research.

5.2. Factors Contributing to Segmentation Performance Gain

The segmentation performance gains achieved by PGPS curricula can be attributed to several factors beyond the improved class balance. For PGPS-Performance, the model is exposed to a larger number of unique classes per iteration, although the correlation between class count and performance was not statistically significant. Additionally, the larger batch sizes in PGPS-Performance increase exposure to data augmentations and promote sampling from more diverse patch locations, leading to broader coverage of the data distribution without losing spatial resolution or altering the forced foreground-to-background ratio. This is supported by the near-significant correlation between patch-to-volume coverage and performance improvement ($\rho \approx -0.4978$, $p \approx 0.0590$) at 50% training length for PGPS-Performance. Interestingly, most correlations are strongest at shorter training lengths. We attribute this to performance saturation at full training, which dampens differences between sampling strategies. Performance gains are consistent with existing studies on LLM training, indicating that shorter input sequences reduce gradient variance, improving training stability [20].

Furthermore, we hypothesize that the aforementioned factors have contributed to the overall performance gain of PGPS over CPS training, but disentangling those factors will be subject to future research. While PGPS-Efficiency cut training runtime strongly, while keeping comparable segmentation performance to CPS, PGPS-Performance was able to significantly outperform standard CPS training on average on 15 datasets, improving segmentation performance on each dataset, while also slightly reducing computational costs.

5.3. PGPS on Different Backbones

PGPS-Performance improves the segmentation performance for all different tested backbones. Therefore, we hypothesize

that limiting the contextual information available to the network, by training on smaller image patches, facilitates more stable and efficient learning, similar to findings reported for large language models in [20]. When trained from scratch, Transformer-based architectures generally lag behind CNNs due to their lack of strong inductive biases [48]. While large-scale pretraining is a common solution to mitigate this limitation [3], several approaches have been proposed to introduce inductive bias directly into Transformers, such as CNN-based teacher distillation [48] or architectural modifications like shifted window attention [49, 4].

Our results demonstrate that the proposed PGPS-Performance curriculum consistently improves segmentation performance across all evaluated architectures, including UNet, UNETR, and SwinUNETR. In contrast, the PGPS-Efficiency variant converges successfully for UNet and SwinUNETR but fails for UNETR. We hypothesize that the stronger inductive biases in UNet and SwinUNETR enable effective learning even under reduced input context, whereas UNETR appears more sensitive to limited information, leading to training instabilities. PGPS-Performance appears to enhance the training signal compared to constant patch size training, thereby improving Dice scores for UNETR as well. Another potential factor for UNETR's instability under PGPS-Efficiency is that simple interpolation of positional embeddings may be suboptimal, while the other two architectures do not rely on such embeddings.

In summary, we hypothesize that training with PGPS-Performance facilitates the learning process by improving the quality of the training signal, particularly for architectures lacking strong inductive bias, such as Transformers. Although we did not explore pretrained Transformer weights in this work, an open research question remains how transfer learning and pretrained representations interact with PGPS-based curricula.

5.4. Comparison to Progressive Resolution

We only compared our PGPS curriculum to the other input-length image curriculum, Progressive Resolution, which is widely used in the Computer Vision domain to efficiently train (foundation) models [24, 25, 26, 30, 31, 27, 28, 29]. Both Progressive Resolution and PGPS are simple by design and do not impose additional computational costs (FLOPs) like hard-negative mining approaches [50, 15] or additional human input [13, 14]. PGPS is automatic and generally applicable to medical image segmentation, and is actually orthogonal to other forms of curricula [15, 50, 13, 14, 16].

Our results demonstrate that PGPS consistently outperforms Progressive Resolution in the tested patch-based segmentation tasks. This is likely due to improved class balancing with PGPS, while Progressive Resolution does not affect the class balance and thereby matches the class balance of CPS. The substantial drop in performance of Progressive Resolution in highly imbalanced tasks further supports this claim. Progressive Resolution is thus only suited for global image-level prediction tasks such as classification [24], image generation [17], contrastive learning [31], and CLIP alignment [30]. By contrast, PGPS is more effective for dense prediction tasks, such as segmentation and potentially object detection, particularly in class-imbalanced scenarios.

5.5. Stochastic Training Variability and Model Validation

A further challenge is the variance in reported performances in repeated training of the same model due to stochastic training variability in deep learning. For example, Dice scores of the default nnU-Net on the BTCV task vary widely, even under the same data splits: 83.37% (ours), 83.76% (MultiTalent Publication [51]), and 83.08% (nnU-Net Revisited [40]). Surprisingly, even the stronger and larger nnU-Net ResEnc variants (M: 83.31%, L: 83.35%, XL: 83.28%) reported in [40] are outperformed by smaller default nnU-Net models reported in the MultiTalent publication [51] and in our experiments in Table 1. While the extent to which sampling alone explains these discrepancies remains unclear, differences in preprocessing (e.g., different sample origin cache in nnU-Net) may also contribute.

Furthermore, to assess the stochastic training variability, we report nnU-Net validation performance from related work that were trained on one of our benchmarking datasets, and compare them to our proposed PGPS framework in Appendix C. We have found that our proposed curriculum resulted in better-performing models than the scores reported in the literature, which were trained via conventional CPS sampling.

In [40], Isensee *et al.* highlight that not all datasets are equally suitable for model comparison due to a high statistical variance (intra-method) or a low systematic (inter-method) variance. Suitable datasets, such as AMOS22, KiTS23, and LiTS, showed higher inter-method than intra-method standard deviation, whereas worse-suited datasets like BTCV showed lower standard deviation in their experiments. Improved convergence via PGPS-Performance yields reduced stochastic difference due to sampling, leading to improved segmentation performance and lower segmentation performance variance as seen in experiment 3.6. This provides a better signal-to-noise ratio, making PGPS-Performance a more reliable strategy for fair model comparison than standard CPS.

Following the strategy in [40], we exemplarily compute inter- and intra-method standard deviation for PGPS-Performance for BTCV and KiTS23. Details on the experiment and results are reported in Appendix D. We found that PGPS-Performance sampling yields better method comparison, as the inter-method to intra-method standard deviation ratio is improved for both datasets when switching from CPS to PGPS-Performance sampling.

5.6. Scalability and Computational Bottlenecks

The primary benefit of PGPS curriculum is resource efficiency. The limit of this speedup is determined by the balance between CPU-driven dataloading and GPU-driven computation. In the PGPS-Efficiency mode, the GPU remains the bottleneck, ensuring a consistent wall-clock speedup over standard constant patch size training because dataloading requirements remain unchanged.

In contrast, the PGPS-Performance mode scales batch sizes extensively, which can shift the bottleneck to the CPU. As investigated in Appendix A, one-patch-per-volume sampling on

large datasets like KiTS23 can double the training time compared to CPS due to I/O overhead. We addressed this by implementing a multi-patch-per-volume sampling strategy, which restores the speed advantage (reducing KiTS23 runtime to 86% of CPS) by sampling multiple patches from a single loaded volume. While framework-level optimizations like subvolume loading (e.g., in recent nnU-Net versions via blosc2 library) can further mitigate these costs, our results suggest that algorithmic sampling strategies remain the most generalizable solution for maintaining scalability across different deep learning libraries.

6. Conclusion and Outlook

In this work, we have introduced a novel curriculum for semantic segmentation based on increasing the patch size during training. We have evaluated the curriculum on 15 diverse and popular 3D medical image segmentation datasets against the default constant patch size baseline. We have proposed two curriculum modes: **PGPS-Performance**, which has been shown to consistently outperform constant patch size sampling on all 15 datasets, in terms of Dice score, while requiring only 90% of the original training time, and **PGPS-Efficiency**, which matches the performance of conventional constant patch size training while drastically reducing training time to 44%. We have demonstrated that modifying the sampling strategy substantially accelerates training convergence and statistically significantly improves final Dice score performance for patch-based 3D medical image segmentation.

We have shown that Progressive Resolution, a well-known input-length automatic curriculum based on image resolution [24, 26, 30, 31, 27, 28, 29], leads to performance decreases in our experiments and is more suited for solving global image-level tasks like classification or contrastive pretraining. This underlines the need and research gap for a curriculum learning approach for dense prediction tasks. In contrast, our proposed curriculum based on patch size did improve the segmentation performance and is, to the best of our knowledge, the first input-length curriculum based on patch size in the computer vision and medical imaging domains.

Our analysis shows that tasks with strong class imbalance benefit most from PGPS, underscoring the importance of sampling strategies in patch-based training. Importantly, the advantages of PGPS are not limited to UNet-style architectures. We have been able to demonstrate consistent improvements for PGPS-Performance across both convolutional and transformer-based models, including UNet, UNETR, and SwinUNETR, highlighting the broad applicability of the method.

A key strength of our approach is its simple integration into any segmentation backbone training. Depending on the application, users can choose between PGPS-Efficiency for rapid experimentation with reduced compute and carbon footprint, or PGPS-Performance for maximizing segmentation performance in deployment settings, while also benefiting from a reduced carbon footprint. Furthermore, PGPS-Performance enables more reliable model comparison by improving the signal-to-noise ratio, due to improved convergence, and thereby providing a clearer signal for methodological benchmarking.

We envisage several promising directions for further research. The curriculum approach sequence length warm-up was shown to reduce the gradient variance in training large language models, enabling larger learning rates and batch sizes, resulting in extremely short training time to only 6% of the original time [20]. We have not explored hyperparameter tuning in our experiments, but anticipate that this could lead to further runtime reduction. Although non-linear patch-size pacing functions could potentially improve curriculum convergence or efficiency, evidence from the NLP domain suggests that such advanced schedules often introduce additional complexity without a corresponding boost in performance [20].

While most research in medical image segmentation is predominantly conducted in the architectural design of backbones, we argue that the sampling strategy itself is a vastly overlooked area and requires more exploration. Advances in differentiable and adaptive sampling strategies, such as differentiable top-k Patch Sampling [38], open new directions for optimizing patch-based sampling. Beyond patch sampling, alternative paradigms such as multi-scale frameworks [37] or cascaded architectures [2] aim to reduce reliance on patching altogether, but patch-based methods remain dominant in segmentation performance today [38].

In summary, we recommend PGPS-Performance as the new default sampling strategy for training 3D patch-based segmentation backbones for deployment. It is conceptually simple, easy to implement within existing frameworks such as nnU-Net, and provides consistent improvements in both segmentation performance and resource efficiency, establishing a strong novel sampling strategy in patch-based 3D medical image segmentation.

Acknowledgements

SMF, JK and JAS acknowledge funding from Munich Center for Machine Learning (MCML). SMF, JCP, and JAS acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 515279324 / SPP 2177. SMF acknowledges funding from the EVUK program (”Next-generation AI for Integrated Diagnostics”) of the Free State of Bavaria. JK and JCP acknowledge funding from the Wilhelm Sander Foundation in Cancer Research (2022.032.1). JK acknowledges funding from the DAAD program Konrad Zuse Schools of Excellence in Reliable Artificial Intelligence, sponsored by the German Federal Ministry of Education and Research. RO and KL acknowledge funding from the EU Horizon Europe and Horizon 2020 research and innovation programme under grant agreement No 101057699 (RadioVal) and No 952103 (EuCanImage), respectively. RO acknowledges a research stay grant from the Helmholtz Information and Data Science Academy (HIDA). JAS and LD are supported by the German Federal Ministry of Research, Technology and Space (DECIPHER-M, 01KD2420G). JAS and DML acknowledge funding from HELMHOLTZ IMAGING, a platform of the Helmholtz Information & Data Science Incubator. SMF and LF

acknowledge funding from the Free State of Bavaria (StMGP) as part of the project 'KI-gestützte multi-modale Diagnostik und stratifizierte Therapie für Endometriose (EndoKI)'.

Data Availability

All data are publicly available. We provide detailed descriptions and proper citations for each dataset used to ensure full transparency and reproducibility. Additionally, all code for data preprocessing and preparation is publicly released and can be used to fully reproduce our results. Our code is publicly available at `https://github.com/compai-lab/2025-MedIA-fischer/releases/tag/v2.6.2`.

Conflicts of Interest

The authors declare that they have no known conflict of interest nor competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work the author(s) used ChatGPT in order to improve the readability of the manuscript. After using this tool, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

CRediT authorship contribution statement

Stefan M. Fischer: Conceptualization, Methodology, Software, Writing - Original Draft, **Johannes Kiechle:** Validation, Writing - Review & Editing, Visualization, **Laura Daza:** Validation, Writing - Review & Editing, **Lina Felsner:** Validation, Writing - Review & Editing, **Richard Osuala:** Validation, Writing - Review & Editing, **Daniel Lang:** Validation, Writing - Review & Editing, **Karim Lekadir:** Validation, Writing - Review & Editing, Funding acquisition, **Jan C. Peeken:** Validation, Writing - Review & Editing, Supervision, Funding acquisition, **Julia A. Schnabel:** Validation, Writing - Review & Editing, Supervision, Funding acquisition

References

[1] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.

[2] F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, K. H. Maier-Hein, nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation, Nature Methods 18 (2) (2021) 203–211.

[3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, in: International Conference on Learning Representations, 2020.

[4] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.

[5] A. Hatamizadeh, Y. Tang, V. Nath, D. Yang, A. Myronenko, B. Landman, H. R. Roth, D. Xu, UNETR: Transformers for 3D medical image segmentation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 574–584.

[6] A. Hatamizadeh, V. Nath, Y. Tang, D. Yang, H. R. Roth, D. Xu, Swin UNETR: Swin transformers for semantic segmentation of brain tumors in MRI images, in: Brainlesion Workshop at International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2021, pp. 272–284.

[7] H. Wang, Q. Jin, S. Li, S. Liu, M. Wang, Z. Song, A comprehensive survey on deep active learning in medical image analysis, Medical Image Analysis 95 (2024) 103201.

[8] L. Yang, Y. Zhang, J. Chen, S. Zhang, D. Z. Chen, Suggestive annotation: A deep active learning framework for biomedical image segmentation, in: International conference on medical image computing and computer-assisted intervention, Springer, 2017, pp. 399–407.

[9] S. U. Saeed, J. Ramalhinho, M. Pinnock, Z. Shen, Y. Fu, N. Montaña-Brown, E. Bonmati, D. C. Barratt, S. P. Pereira, B. Davidson, et al., Active learning using adaptable task-based prioritisation, Medical Image Analysis 95 (2024) 103181.

[10] S. Czolbe, K. Arnavaz, O. Krause, A. Feragen, Is segmentation uncertainty useful?, in: International conference on information processing in medical imaging, Springer, 2021, pp. 715–726.

[11] Y. Bengio, J. Louradour, R. Collobert, J. Weston, Curriculum learning, in: Proceedings of the International Conference on Machine Learning, PMLR, 2009, pp. 41–48.

[12] R. Selvan, N. Bhagwat, L. F. Wolff Anthony, B. Kanding, E. B. Dam, Carbon footprint of selecting and training deep learning models for medical image analysis, in: Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2022, pp. 506–516.

[13] A. Jiménez-Sánchez, D. Mateus, S. Kirchhoff, C. Kirchhoff, P. Biberthaler, N. Navab, M. A. González Ballester, G. Piella, Medical-based deep curriculum learning for improved fracture classification, in: Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2019, pp. 694–702.

[14] J. Wei, A. Suriawinata, B. Ren, X. Liu, M. Lisovsky, L. Vaickus, C. Brown, M. Baker, M. Nasir-Moin, N. Tomita, L. Torresani, J. Wei, S. Hassanpour, Learn like a pathologist: Curriculum learning by annotator agreement for histopathology image classification, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 2473–2483.

[15] A. Jesson, N. Guizard, S. H. Ghalehjegh, D. Goblot, F. Soudan, N. Chapados, Cased: Curriculum adaptive sampling for extreme data imbalance, in: Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2017, pp. 639–646.

[16] M. Havaei, N. Guizard, N. Chapados, Y. Bengio, HeMIS: Hetero-modal image segmentation, in: Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2016, pp. 469–477.

[17] T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of gans for improved quality, stability, and variation, in: International Conference on Learning Representations, 2018.

[18] J. Zhao, L. Dai, M. Zhang, F. Yu, M. Li, H. Li, W. Wang, L. Zhang, PGU-net+: Progressive growing of U-net+ for automated cervical nuclei segmentation, in: Multiscale Multimodal Medical Imaging Workshop at International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2020, pp. 51–58.

[19] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), 2019, pp. 4171–4186.

[20] C. Li, M. Zhang, Y. He, The stability-efficiency dilemma: Investigating sequence length warmup for training gpt models, Advances in Neural Information Processing Systems 35 (2022) 26736–26750.

[21] O. Press, N. A. Smith, M. Lewis, Shortformer: Better language model-

ing using shorter inputs, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2021, pp. 5493–5505.

[22] L. Shen, Y. Sun, Z. Yu, L. Ding, X. Tian, D. Tao, On efficient training of large-scale deep learning models, ACM Computing Surveys 57 (3) (2024) 1–36.

[23] C. Li, Z. Yao, X. Wu, M. Zhang, C. Holmes, C. Li, Y. He, Deepspeed data efficiency: Improving deep learning model quality and training efficiency via efficient data sampling and routing, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, 2024, pp. 18490–18498.

[24] M. Tan, Q. Le, Efficientnetv2: Smaller models and faster training, in: International conference on machine learning, PMLR, 2021, pp. 10096–10106.

[25] Y. Wang, Y. Yue, R. Lu, Y. Han, S. Song, G. Huang, Efficienttrain++: Generalized curriculum learning for efficient visual backbone training, IEEE Transactions on Pattern Analysis and Machine Intelligence (2024).

[26] D. Bolya, P.-Y. Huang, P. Sun, J. H. Cho, A. Madotto, C. Wei, T. Ma, J. Zhi, J. Rajasegaran, H. Rasheed, J. Wang, M. Monteiro, H. Xu, S. Dong, N. Ravi, D. Li, P. Dollár, C. Feichtenhofer, Perception encoder: The best visual embeddings are not at the output of the network, arXiv:2504.13181 (2025).

[27] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al., Dinov2: Learning robust visual features without supervision, arXiv preprint arXiv:2304.07193.

[28] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, et al., Dinov3, arXiv preprint arXiv:2508.10104.

[29] M. Assran, A. Bardes, D. Fan, Q. Garrido, R. Howes, M. Muckley, A. Rizvi, C. Roberts, K. Sinha, A. Zholus, et al., V-jepa 2: Self-supervised video models enable understanding, prediction and planning, arXiv preprint arXiv:2506.09985.

[30] R. Li, D. Kim, B. Bhanu, W. Kuo, Reclip: Resource-efficient clip by training with small images, Transactions on Machine Learning Research.

[31] M. T. Koçyiğit, T. M. Hospedales, H. Bilen, Accelerating self-supervised learning via efficient training strategies, in: Proceedings of the IEEE/CVF winter conference on applications of computer vision, 2023, pp. 5654–5664.

[32] E. Arani, S. Marzban, A. Pata, B. Zonooz, Rgpnet: A real-time general purpose semantic segmentation, in: Proceedings of the IEEE/CVF winter conference on applications of computer vision, 2021, pp. 3009–3018.

[33] S. M. Fischer, L. Felsner, R. Osuala, J. Kiechle, D. M. Lang, J. C. Peeken, J. A. Schnabel, Progressive growing of patch size: Resource-efficient curriculum learning for dense prediction tasks, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2024, pp. 510–520.

[34] S. Roy, G. Koehler, C. Ulrich, M. Baumgartner, J. Petersen, F. Isensee, P. F. Jaeger, K. H. Maier-Hein, Mednext: transformer-driven scaling of convnets for medical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2023, pp. 405–415.

[35] J. Ma, F. Li, B. Wang, U-mamba: Enhancing long-range dependency for biomedical image segmentation, arXiv preprint arXiv:2401.04722 (2024).

[36] F. Isensee, P. F. Jäger, S. A. Kohl, J. Petersen, K. H. Maier-Hein, Automated design of deep learning methods for biomedical image segmentation, arXiv preprint arXiv:1904.08128 (2019).

[37] K. Kamnitsas, C. Ledig, V. F. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, B. Glocker, Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation, Medical Image Analysis 36 (2017) 61–78. doi:https://doi.org/10.1016/j.media.2016.10.004.

[38] Y. S. Jeon, H. Yang, H. Fu, Y. Kway, M. Feng, No more sliding window: Efficient 3d medical image segmentation with differentiable top-k patch sampling, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2025, pp. 376–386.

[39] M. J. Cardoso, W. Li, R. Brown, N. Ma, E. Kerfoot, Y. Wang, B. Murrey, A. Myronenko, C. Zhao, D. Yang, et al., Monai: An open-source framework for deep learning in healthcare, arXiv preprint arXiv:2211.02701.

[40] F. Isensee, T. Wald, C. Ulrich, M. Baumgartner, S. Roy, K. Maier-Hein, P. F. Jaeger, nnu-net revisited: A call for rigorous validation in 3d medical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2024, pp. 488–498.

[41] M. Antonelli, A. Reinke, S. Bakas, K. Farahani, A. Kopp-Schneider, B. A. Landman, G. Litjens, B. Menze, O. Ronneberger, R. M. Summers, et al., The medical segmentation decathlon, Nature Communications 13 (1) (2022) 4128. doi:10.1038/s41467-022-30695-9.

[42] F. Bolelli, K. Marchesini, N. van Nistelrooij, L. Lumetti, V. Pipoli, E. Ficarra, S. Vinayahalingam, C. Grana, Segmenting maxillofacial structures in cbct volumes, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2025, pp. 5238–5248.

[43] J. Wasserthal, H.-C. Breit, M. T. Meyer, M. Pradella, D. Hinck, A. W. Sauter, T. Heye, D. T. Boll, J. Cyriac, S. Yang, et al., Totalsegmentator: robust segmentation of 104 anatomic structures in ct images, Radiology: Artificial Intelligence 5 (5) (2023) e230024.

[44] N. Heller, A. Wood, F. Isensee, T. Rädsch, R. Teipaul, N. Papanikolopoulos, C. Weight, Kidney and Kidney Tumor Segmentation: MICCAI 2023 Challenge, KiTS 2023, Held in Conjunction with MICCAI 2023, Vancouver, BC, Canada, October 8, 2023, Proceedings, Vol. 14540, Springer Nature, 2024.

[45] Y. Ji, H. Bai, C. Ge, J. Yang, Y. Zhu, R. Zhang, Z. Li, L. Zhanng, W. Ma, X. Wan, et al., Amos: A large-scale abdominal multi-organ benchmark for versatile medical image segmentation, Advances in neural information processing systems 35 (2022) 36722–36732.

[46] B. Landman, Z. Xu, J. Iglesias, M. Styner, T. Langerak, A. Klein, Miccai multi-atlas labeling beyond the cranial vault–workshop and challenge, in: Proceedings of the MICCAI Multi-Atlas Labeling Beyond Cranial Vault Workshop Challenge, 2015.

[47] F. Isensee, Y. Kirchhoff, L. Kraemer, M. Rokuss, C. Ulrich, K. H. Maier-Hein, Scaling nnu-net for cbct segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2024, pp. 13–20.

[48] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, in: International conference on machine learning, PMLR, 2021, pp. 10347–10357.

[49] S. H. Lee, S. Lee, B. C. Song, Vision transformer for small-size datasets, arXiv preprint arXiv:2112.13492 (2021).

[50] A. Shrivastava, A. Gupta, R. Girshick, Training region-based object detectors with online hard example mining, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 761–769.

[51] C. Ulrich, F. Isensee, T. Wald, M. Zenk, M. Baumgartner, K. H. Maier-Hein, Multitalent: A multi-dataset approach to medical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2023, pp. 648–658.

[52] T. Wald, S. Roy, F. Isensee, C. Ulrich, S. Ziegler, D. Trofimova, R. Stock, M. Baumgartner, G. Köhler, K. Maier-Hein, Primus: Enforcing attention usage for 3d medical image segmentation, arXiv preprint arXiv:2503.01835.

[53] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, K. H. Maier-Hein, nnu-net: a self-configuring method for deep learning-based biomedical image segmentation, `https://github.com/MIC-DKFZ/nnUNet`, accessed: 2025-10-08 (2024).

[54] G. Koehler, T. Wald, C. Ulrich, D. Zimmerer, P. F. Jaeger, J. K. Franke, S. Kohl, F. Isensee, K. H. Maier-Hein, Recyclenet: Latent feature recycling leads to iterative decision refinement, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 810–818.

[55] F. Isensee, C. Ulrich, T. Wald, K. H. Maier-Hein, Extending nnu-net is all you need, in: BVM workshop, Springer, 2023, pp. 12–17.

[56] T. Wald, I. E. Hamamci, Y. Gao, S. Bond-Taylor, H. Sharma, M. Ilse, C. Lo, O. Melnichenko, A. Schwaighofer, N. C. Codella, et al., Comprehensive language-image pre-training for 3d medical image understanding, arXiv preprint arXiv:2510.15042 (2025).

[57] J. Becktepe, L. Hennig, S. Oeltze-Jafra, M. Lindauer, Auto-nnu-net: Towards automated medical image segmentation, arXiv preprint arXiv:2505.16561 (2025).

Appendix A. Batch Construction Strategies

Setup: To efficiently utilize GPU resources during early PGPS-Performance stages with smaller patches, the batch size was increased either by (1) drawing patches from more patient volumes or (2) extracting multiple patches per patient volume. We investigated four patch sampling strategies, *Single-Volume*, *Multi-Crop*, *Split-Crop*, and *Single-Crop*, on the MSD Lung Tumor dataset to analyze their influence on convergence and generalization.

Batching Strategies: *Single-Volume* crops all patches from one patient volume, while *Multi-Crop* and *Split-Crop* sample from two patient volumes. In *Split-Crop*, one patient provides foreground patches and the other background patches, whereas *Multi-Crop* draws both from each patient. *Single-Crop* follows the nnU-Net scheme, loading as many patients as the batch size, drawing one patch per patient, thereby maximizing patient diversity, preventing patch overlaps.

Results on MSD Lung: All PGPS-Performance batching strategies converged faster than the baseline Constant Patch Size (CPS) setup (Fig. A.6). *Single-Volume* exhibited strong overfitting, and *Multi-Crop* showed moderate overfitting. *Split-Crop* achieved the most stable convergence and highest Dice performance, while *Single-Crop* slightly lagged in convergence speed. Runtime analysis showed that *Split-CropSplit-Crop* and *Multi-Crop* ran efficiently (82% of CPS runtime), whereas *Single-Crop* required more dataloading time (87% of CPS runtime) because for each batch, many distinct patient volumes have to be loaded.

Cross-Dataset Runtime Comparison: While *Split-Crop* allows patch overlaps, *Single-Crop* prevents this. To quantify the effect on large high-quality benchmark datasets, we pick AMOS22 and KiTS23 to evaluate segmentation performance and computational costs. Table .4 reports Dice scores, FLOPs, and runtime normalized to the CPS baseline. Both PGPS variants consistently improved segmentation performance over CPS, but *Split-Crop* was much more efficient. *Single-Crop* runtime increased drastically, 205% on KiTS23 and 198% on AMOS22, caused by extensive I/O from loading large numbers of patient volumes. In contrast, *Split-Crop* achieved nearly identical performance with far lower computational overhead (82–87% of CPS runtime), while achieving higher segmentation performance than *Single-Crop* in AMOS22 and MSD Lung. For larger-batch datasets such as TotalSegmentatorV2 (batch size 1575), these runtime penalties for *Single-Crop* would become prohibitively large.

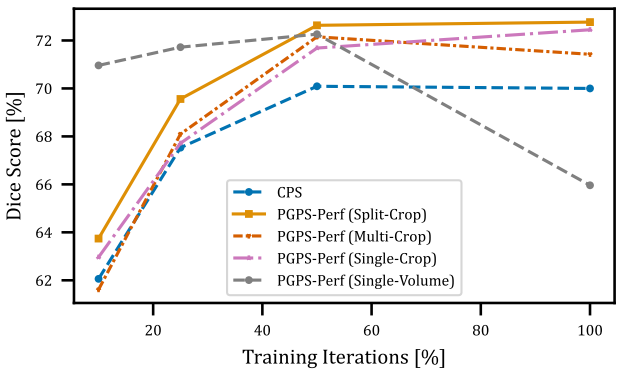


Figure A.6: Convergence of different batching strategies for PGPS-Performance on the MSD Lung Tumor dataset. Increasing batch size can be achieved by sampling from more patients or generating multiple crops per patient. Overfitting occurs when both foreground and background patches are drawn from the same patient.

Conclusion: The choice of batch construction has a strong impact on generalization and computational efficiency. *Split-Crop* offers the best trade-off between convergence speed, segmentation performance, and runtime, and was therefore adopted as the default batching strategy for all PGPS-Performance experiments. The *Single-Crop* strategy remains useful for applications prioritizing maximal patch variety. Enlarging patch diversity by preventing patch overlaps did not seem to improve performance in our experiments.

Our experiments were run with nnU-Net version 2.2.0, except for the newer ResUNet model experiment in Appendix D, for which we used nnU-Net version 2.6.1. In the older nnU-Net versions, full volumes are loaded to create patches, resulting in high dataloading costs. For that case, loading many volumes can significantly increase the training wall-clock time, as dataloading becomes the bottleneck of the training pipeline, as seen for AMOS22 and KiTS23 training with *Single-Crop* strategy. Newer versions of nnU-Net (from 2.6 on) enable subvolume loading via the *blosc2* data caching and thus reduce the dataloading costs heavily. This also allows us to run the *Single-Crop* strategy with shorter wall-clock runtimes than CPS for the new nnU-Net versions.

Furthermore, we argue that there is overfitting for the *Multi-Crop* and *Single-Volume* strategy, which might stem from a lower batch diversity. *Single-Volume* offers the lowest diversity due to the highest overlap of patches. On the other side, *Single-Crop* potentially also leads to overfitting due to seeing all lesions in the dataset during the small patch size stages. *Split-Crop*, overcomes both potential risks.

Appendix B. Orientation Sensitivity

Setup: Smaller patches reduce global spatial context, which can hinder segmentation of directionally defined structures (e.g., left/right or upper/lower). Orientation sensitivity varies across datasets: BTCV and AMOS22 each include 4 directionally defined classes out of 14 and 15, TotalSegmentatorV2 includes 66 out of 107, and ToothFairy2 includes 32 out of 42. Mirror augmentation, the nnU-Net default, can further obscure orientation cues by inverting anatomical structures. We trained CPS and PGPS-Performance with and without mirror augmentation on these datasets using 5-fold cross-validation.

Results: As shown in Table B.5, the influence of mirror augmentation depends strongly on dataset orientation sensitivity. For BTCV and AMOS22, effects were minimal, with PGPS-Performance consistently outperforming CPS. In TotalSegmentatorV2, disabling mirroring substantially improved results, and PGPS-Performance achieved the best overall Dice score.

Table .4: Comparison of Dice scores, FLOPs, and relative runtime across three datasets for Constant Patch Size (CPS) and Progressive Growing of Patch Size (PGPS) variants with split- and single-crop sampling. All values are normalized to the CPS baseline. PGPS-Perf (Single-Crop) also improves Dice score over CPS but incurs substantial runtime overhead due to increased dataloading costs.

Metric	MSD6 Lung Tumor			KiTS23			AMOS22		
	CPS	PGPS-Perf (Split-Crop)	PGPS-Perf (Single-Crop)	CPS	PGPS-Perf (Split-Crop)	PGPS-Perf (Single-Crop)	CPS	PGPS-Perf (Split-Crop)	PGPS-Perf (Single-Crop)
Dice score [%] ↑	70.00	**72.77**	72.45	86.02	86.46	**87.18**	88.62	**88.78**	88.77
Rel. FLOPs [%] ↓	100	**89**	**89**	100	**84**	**84**	100	**89**	**89**
Rel. Runtime [%] ↓	100	**82**	87	100	**86**	205	100	**87**	198

The strongest degradation occurred in ToothFairy2, where most classes are directionally defined; mirror augmentation severely reduced performance for CPS and PGPS-Performance, but particularly for the curriculum. Disabling mirroring did result in a large performance gain for both sampling strategies in ToothFairy2, where PGPS-Performance did outperform CPS.

Overall, PGPS-Performance remains beneficial when orientation-sensitive classes form only a subset of labels, but can suffer when such classes dominate the dataset. We therefore recommend caution with mirror augmentation in tasks with large numbers of directional-defined classes, as it may conflict with PGPS's progressive curriculum and limit performance gains. Additionally, a CPS-trained network would also benefit from disabling mirroring.

Table B.5: Impact of mirror augmentation on CPS and PGPS across datasets containing directionally defined semantic classes. Results are mean Dice scores (%) over 5-fold cross-validation.

Dataset	With Mirror		Without Mirror	
	CPS	PGPS-Perf	CPS	PGPS-Perf
BTCV	83.37	**83.81**	82.93	**83.09**
AMOS22	88.62	**88.78**	88.56	**88.63**
TotalSegV2	84.56	**84.64**	87.82	**88.15**
ToothFairy2	**63.23**	47.08	76.92	**77.00**

Appendix C. nnU-Net Performance in Related Work

Setup: We assessed the variability in segmentation performance that was reported for nnU-Net baselines in the literature. Therefore, we reviewed publications that employed nnU-Net in their studies and reported results from 5-fold cross-validation on datasets contained in our benchmark.

Results: In total, we identified ten sources (nine publications and the official DKFZ nnU-Netv2 GitHub repository) that provided relevant validation scores. Although this collection might not be exhaustive, it reflects the diversity of nnU-Net performance scores currently published in the field.

Table C.6 summarizes Dice scores reported across the Medical Segmentation Decathlon (MSD), BTCV, KiTS23, and AMOS22 datasets. Most of these results originate from the DKFZ research group [34, 36, 40, 53, 54, 55], who developed and maintain nnU-Net framework. Another external publication was found using nnU-Net as baseline [57]. In Figure C.7, we plot the normalized standard deviation in performance over all trained models per dataset. Note that scores originate from different nnU-Net versions (nnU-Netv1, nnU-Netv2 and different pip package versions), but DKFZ claims that both maintain equivalent segmentation performance [53].

For datasets, where we found multiple CPS-based results, as seen in Table C.6, the variance in reported performance underscores the variability of training outcomes across different runs. Despite this variability, our proposed PGPS-Performance curriculum consistently achieves the highest Dice score across all evaluated datasets except MSD2. The exception for MSD2 could result from orientation sensitivity, described in Appendix B, as for MSD2, only the left atrium has to be segmented.

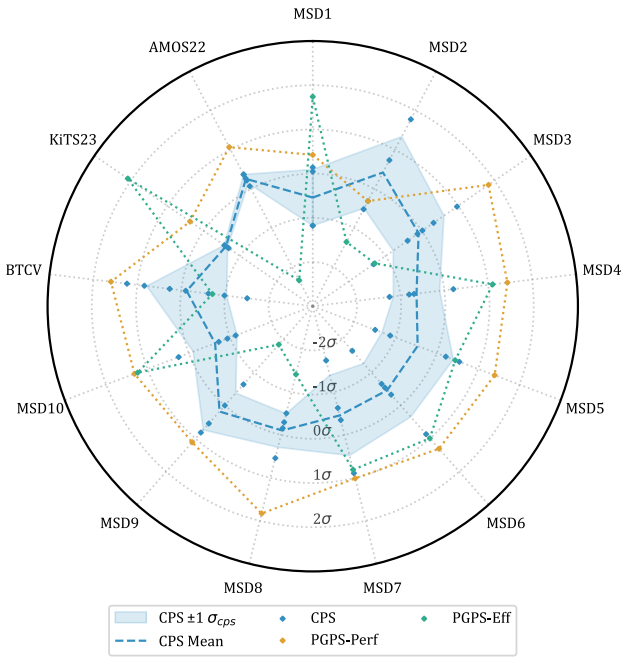


Figure C.7: Radar Plot of publicly available vanilla nnU-Net segmentation performances, trained via standard constant patch size, relative to our PGPS curricula results. We standardized Dice scores across all training runs per dataset using z-score normalization. Training with PGPS-Performance consistently outperformed CPS across all datasets, except MSD2, while PGPS-Efficiency was on par with CPS. Performance values per publication are shown in Table C.6.

Table C.6: Comparison of nnU-Net baselines and related models across the Medical Segmentation Decathlon (MSD), BTCV, KiTS23, and AMOS22-CT/MRI datasets. Reported values are the mean Dice scores over 5-fold cross-validation as provided in the respective publications. The final two rows show the performance with our proposed **Progressive Growing of Patch Size** curriculum, while all other rows show nnU-Net instances trained using the default **constant patch size (CPS)** strategy. Bold values indicate the best result per dataset, underlined values the second best. Overall, models trained via our proposed PGPS-Performance curriculum yield, in all scenarios except MSD2, better segmentation performance than the CPS baseline. In two scenarios, it PGPS-Efficiency scored first before PGPS-Performance. Note that for PRIMUS [52], the authors only averaged over 4 instead of 5 folds in the original publication.

Original Publication	MSD1	MSD2	MSD3	MSD4	MSD5	MSD6	MSD7	MSD8	MSD9	MSD10	BTCV	KiTS23	AMOS22-CT/MRI
Constant Patch Size:													
nnU-Net (orig.) [36]	74.11	93.28	79.71	88.91	75.37	72.11	67.45	68.37	96.38	45.53	82.79	–	–
nnU-Net v2 (Repo) [53]	73.98	93.34	79.53	88.95	75.01	69.83	67.17	68.41	96.90	45.04	83.25	–	–
nnU-Net revisited [40]	–	–	–	–	–	–	–	–	–	–	83.08	86.04	88.64
MultiTalent [51]	–	–	–	–	–	–	–	–	–	–	83.76	–	–
MedNeXt [34]	–	–	–	–	–	–	–	–	–	–	83.56	–	–
RecycleNet [54]	–	–	–	–	–	–	–	–	–	–	82.96	–	88.58
Extending nnU-Net is All You Need [55]	–	–	–	–	–	–	–	–	–	–	–	–	88.64
COLIPRI [56]	–	–	80.09	–	–	70.32	–	–	–	–	–	86.04	–
Primus [52]	–	–	79.29	–	–	–	–	–	–	–	–	85.99	88.61
Auto-nnU-Net [57]	73.98	**93.39**	79.45	89.04	73.53	68.33	66.07	68.31	96.66	46.04	–	–	–
Ours (CPS)	74.12	93.29	78.74	88.96	73.13	70.00	68.68	68.61	97.02	48.41	83.37	86.02	88.62
Progressive Growing of Patch Size:													
Ours (PGPS-Efficiency)	**74.29**	93.24	78.76	89.12	75.26	72.30	68.60	68.05	95.85	50.83	83.05	**87.22**	88.10
Ours (PGPS-Performance)	74.15	93.29	**80.60**	**89.15**	**76.31**	**72.77**	**68.80**	**68.98**	**97.15**	**51.02**	**83.81**	86.46	**88.78**

Overall, these findings confirm that our proposed PGPS-Performance does result in improved performance over repeated runs on different hardware and environments.

Appendix D. More Reliable Backbone Comparison

Setup: As discussed in Section 5.5, the proposed PGPS-Performance strategy is expected to provide a more reliable basis for model comparison than standard Constant Patch Size (CPS) training. To evaluate this assumption, we conducted experiments on the BTCV and KiTS23 datasets, which were identified by [40] as the least and most reliable benchmarks for backbone comparison, respectively. Following [40], we compute the ratio between the inter-method and intra-method standard deviations. A higher ratio indicates that architectural differences dominate over random variation, meaning that the dataset and training setup allow for more meaningful model comparison. We used the nnU-Net ResEnc variants (M, L, and XL), which scale both patch size and model capacity, alongside the original nnU-Net configuration. Each model was trained using both CPS and PGPS-Performance sampling strategies, allowing us to assess whether PGPS-Performance improves the signal-to-noise ratio in backbone comparisons.

Table D.7: Comparison of Dice scores between Constant Patch Size (CPS) and Progressive Growing of Patch Size (PGPS) training strategies across nnU-Net ResEnc variants on BTCV and KiTS23 datasets. Values indicate mean Dice scores over 5-fold cross-validation. Bold values mark the highest performance per architecture and dataset.

Model	**BTCV**		**KiTS23**	
	CPS	PGPS-Perf	CPS	PGPS-Perf
nnU-Net (orig.)	83.37	**83.81**	86.02	**86.46**
ResEncM	83.48	**83.97**	87.05	**88.00**
ResEncL	83.30	**84.18**	88.24	**88.64**
ResEncXL	83.30	**84.03**	88.68	**88.93**

Table D.8: Comparison of Dice score variability across nnU-Net and ResEnc variants for Constant Patch Size (CPS) and Progressive Growing of Patch Size (PGPS) on BTCV and KiTS23. Reported values represent the mean standard deviation of the Dice score over 5-fold cross-validation. The inter/intra-method ratio indicates how clearly the dataset ranks architectures (higher is better). PGPS-Performance increases this ratio, improving backbone comparability.

	BTCV		**KiTS23**	
	CPS	**PGPS-Perf**	**CPS**	**PGPS-Perf**
nnU-Net (orig.)	2.24	2.54	1.77	1.73
ResEncM	2.09	2.25	1.55	1.40
ResEncL	2.52	2.48	0.92	0.94
ResEncXL	2.20	2.27	1.19	0.91
Avg. Intra-Method SD [%]	2.27	2.38	1.358	1.245
Inter-Method SD [%]	0.74	1.33	1.037	0.955
Inter/Intra Ratio [%] ↑	32.6	**55.6**	76.4	**76.7**

Results: Table D.7 summarizes mean Dice scores across both datasets. For BTCV, the lowest performing nnU-Net variant trained via PGPS-Performance resulted in even higher segmentation performance than all model variants for CPS sampling. Also for KiTS23, PGPS-Performance sampling improved segmentation performance over CPS sampling, while increasing model capacity had a higher impact on the segmentation performance for all variants.

While PGPS-Performance generally improved absolute performance for all architectures, another critical finding concerns the reduction in statistical variance across training runs and architectures. Table D.8 shows the intra- and inter-method standard deviations and the resulting inter/intra ratios. On the KiTS23 dataset, previously shown to provide the very stable comparison signal [40], the ratio remains nearly constant (76.4% vs. 76.7%), indicating that both CPS and PGPS-Performance allow for consistent differentiation of model scales. In contrast, on BTCV, which is known for a low signal-to-noise ratio for backbone comparison, PGPS-Performance substantially increased the ratio from 32.6% to

Table D.9: Comparison of Dice scores [%] across the Medical Segmentation Decathlon (MSD) tasks. We report results for the baseline constant patch size training (CPS), the **new PGPS modes** introduced in this work (PGPS-Eff, PGPS-Perf), and the **original PGPS modes** from MICCAI 2024 (PGPS, PGPS+) [33]. Values are color-coded relative to CPS: green = better than CPS, red = worse than CPS. **Bold**: Best performing sampling. Underlined: Second best performing sampling. All values are mean Dice scores in 5-fold cross-validation as in [36]. The newly proposed PGPS-Efficiency is more computationally efficient than the old version, while also yielding a better Dice score. PGPS-Performance achieves the highest Dice score and is the only curriculum outperforming CPS on every task, while it has doubled the computational costs compared to PGPS+. While PGPS-Efficiency is the most efficient version of the four curricula, PGPS-Performance yields the highest segmentation performance.

Dataset	CPS	New (This Work)		Old (MICCAI 2024)	
		PGPS-Eff	PGPS-Perf	PGPS	PGPS+
MSD Brain	74.12	**74.29**	74.15	74.12	74.21
MSD Heart	93.29	93.24	**93.29**	93.21	93.28
MSD Liver	78.74	78.76	**80.60**	78.91	79.38
MSD Hippocampus	88.96	89.12	**89.15**	89.11	89.07
MSD Prostate	73.13	75.26	**76.31**	75.66	75.31
MSD Lung	70.00	72.30	72.77	72.63	**73.33**
MSD Pancreas	68.68	68.60	**68.80**	68.24	68.22
MSD Hepatic Vessel	68.61	68.05	**68.98**	67.82	68.71
MSD Spleen	97.02	95.85	**97.15**	96.21	96.54
MSD Colon	48.41	50.83	**51.02**	49.25	49.67
Avg. Normalized Dice Score [↑]	100.00	100.94	**101.72**	100.66	101.04
Rel. FLOPs [↓]	100.00	**0.32**	0.88	0.35	0.41

55.6%. This demonstrates that PGPS-Performance effectively increases the signal-to-noise ratio, thereby yielding more reliable and interpretable backbone comparisons. Intra-Method SD is only decreased for KiTS23, while for BTCV it is increased. In our experiment in section 3.6, we focused on repeating training on the same fold. In contrast, here the standard deviation is computed over five folds, such that fold difficulty (annotation errors, sample difficulty) itself also plays an important role.

Overall, these findings confirm that PGPS-Performance enhances segmentation performance but also improves the robustness of studies comparing different methods or backbones.

Appendix E. Comparison Between MICCAI 2024 PGPS Modes and Newly Proposed PGPS Modes

This section compares the PGPS modes introduced in our MICCAI 2024 work (**PGPS, PGPS+**) [33] with the newly proposed variants (**PGPS-Efficiency, PGPS-Performance**). The aim is to highlight methodological updates and their practical effects.

Appendix E.1. Methodological Differences

The differences between the MICCAI 2024 and current PGPS modes are:

- **Additional patch size stage:** The new modes code we disabled a PyTorch normalization layer check that previously blocked single-element tensors, enabling an extra patch size stage. This results in slightly higher class balance for both new PGPS curricula and larger batch sizes for PGPS-Performance.
- **Patch size increments:** Instead of fixed sequential axis growth (MICCAI 2024), the new modes always expand the smallest axis first, leading to more balanced spatial scaling and larger average batch sizes.
- **Batch creation strategy:** PGPS+ relies on *Single-Crop*, whereas PGPS-Performance adopts a *Split-Crop* strategy. This enables efficient training with much larger batch sizes by reducing dataloading costs.
- **Batch size policy:** PGPS+ increases batch size such that the input tensor dimensions grow monotonically, while PGPS-Performance fully utilizes GPU memory, increasing the average batch size and class balance.

Appendix E.2. Results and Observations

Table D.9 summarizes Dice scores and relative computational costs for the Medical Segmentation Decathlon tasks. Key observations include:

- **Efficiency:** PGPS-Efficiency reduces relative FLOPs compared to PGPS while achieving slightly higher Dice scores, benefiting from the additional patch size stage and improved class balance.
- **Performance:** PGPS-Performance achieves the highest overall segmentation performance, outperforming CPS on every task and exceeding PGPS+ due to its larger batch size and class balance.

Overall, the newly proposed PGPS modes mark a substantial improvement over the MICCAI 2024 versions. PGPS-Efficiency provides the best trade-off between segmentation performance and computational cost. PGPS-Performance delivers the highest segmentation performance while still reducing computational costs compared to CPS. Crucially, the *Split-Crop* strategy in PGPS-Performance drastically reduces dataloading costs, making large-batch 3D segmentation training

feasible, whereas *Single-Crop* used in PGPS+ would be prohibitively expensive for large batch sizes in terms of dataloading costs, as seen in Table .4 for datasets KiTS23 and AMOS22, yielding training times double as long as for CPS.

Algorithm 1 Progressive Growing of Patch Size within the nnUNet Framework. Models are trained on progressively increasing patch sizes, starting from a small initial patch size (P_{min}) and reaching a final patch size (P_{max}) used also for inference. With PGPS the class balance during training is improved beyond just oversampling foreground patches. A curriculum is governed by a pacing function $f(t)$, which maps each training iteration t to a patch size, starting at patch size P_{min} and finishing at patch size P_{max}. A batch sampling policy μ regulates the ratio of foreground oversampling and the number of patches created per volume. PGPS offers two operating modes: Resource-efficient mode maintains a constant batch size (b_{max}) to reduce wall-clock time through fewer FLOPs; Performance mode scales the batch size inversely with patch volume to maximize segmentation performance while keeping total FLOPs comparable to standard training. nnU-Net as a framework configures many hyperparameters by heuristic rules. The "numel" operation counts the number of elements in the tensor. The complete code is given on GitHub at `https://github.com/compai-lab/2025-MedIA-fischer/releases/tag/v2.6.2`

Require:

M: Model Architecture (configured by nnU-Net)

P_{min}: minimal patch size that architecture can process (architecture-dependent, thus configured by nnU-Net): For nnU-Net's default UNet, the minimum patch size length depends on the number of downsampling operations, and is given by 2^{num_pool}, where *num_pool* is the number of pooling operations for each axis. For the tested Transformer-based variants (UNETR or SwinUNETR): The minimum patch size is the internal transformer token size

P_{max}: terminal patch size (configured by nnU-Net)

b_{base}: terminal batch size (configured by nnU-Net)

T: total number of training iterations (configured by nnU-Net)

$m \in$ {resource-efficient, performance}: curriculum mode

$f : \mathbb{N} \to \mathbb{N}$: pacing function, $f(1) = P_{\text{min}}$, $f(T) = P_{\text{max}}$; Implemented as a simple step-wise linear function of patch size over training iterations, where step sizes are defined by model architecture, which can only process multiples of P_{min}. To keep task similarity between patch sizes as large as possible, we increase one patch size axis at a time and always increase the patch size axis that is currently the smallest axis. Each patch size stage (P_{min}, ..., P_{max}) is trained for the same number of iterations.

μ: batch sampling policy; Implemented by sampling b_{base} random volumes. Forcing $\frac{b_{\text{base}}}{2}$ foreground patches from first half of the sampled volumes and $\frac{b_{\text{base}}}{2}$ random patches from second half of the sampled volumes. nnU-Net generates foreground patches by selecting a foreground voxel and using this as a center for the new foreground patch.

1: **for** iteration $t = 1$ **to** T **do**

2: Retrieve current patch size: $P \leftarrow f(t)$

3: Compute batch size:

$$b \leftarrow \begin{cases} \left\lfloor \dfrac{\text{numel}(b_{\text{base}}, P_{\text{max}})}{\text{numel}(P)} \right\rfloor \cdot b_{\text{base}} & \text{if } m = \text{performance} \\ b_{\text{base}} & \text{if } m = \text{resource-efficient} \end{cases}$$

4: Sample a mini-batch of b patches of size P according to policy μ

5: Perform forward pass, compute loss, update model weights

6: **end for**

7: **Inference:** Use terminal patch size P_{max} for inference

Declaration of interests

☐ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Given her role as Associate Editor, Julia A. Schnabel had no involvement in the peer review of this article and had no access to information regarding its peer review. Full responsibility for the editorial process for this article was delegated to another journal editor. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.